Universidad Carlos III de Madrid

Escuela Politécnica Superior

Grado en Ingeniería en Tecnologías Industriales

Trabajo de Fin de Grado

Diseño y construcción de un robot móvil con navegación controlada a través de una aplicación móvil con interfaz de voz

Autor: Jesús Vicente López

Tutor: Jonathan Crespo Herrero

Septiembre, 2014

A mis padres y a mi hermana por estar siempre ahí,

a mi novia porque la quiero mucho,

a mis amigos porque me quieren mucho

y a mi tutor por haberme ayudado a realizarlo.

Resumen

En este trabajo se pretende abarcar el diseño y la fabricación de un robot móvil con la intención de estudiar la viabilidad de fabricarlo con un bajo coste pero con capacidades de navegación autónoma y reconocimiento del entorno.

El trabajo es amplio y toca diferentes aspectos relacionados con la ingeniería y la robótica ya que abarca desde el diseño mecánico de la plataforma hasta la programación del microcontrolador que lo dirige. A medida que el lector vaya avanzando se irá dando cuenta de que la estructura del trabajo sigue este mismo orden con el objetivo de facilitar su lectura.

El robot que aquí se presenta está orientado en el ámbito de la robótica educacional ya que en su diseño se han utilizado diferentes tecnologías Open-Source, como Arduino y App Inventor. La intención del autor es compartir este trabajo, al igual que la mayoría de las personas que trabajan con estas tecnologías comparten los suyos.

Como aspecto original e innovador a destacar, el robot es controlado por comandos de voz a través de una aplicación móvil expresamente diseñada y programada para ello. Esta herramienta permite al usuario controlar al robot pulsando solamente un botón y diciéndole el comando de voz requerido al teléfono sin importar la ubicación del robot en ese instante. Además puede ver representada su trayectoria por pantalla y las medidas de sus sensores.

Abstract

In this Project it is intended to cover the design and fabrication process of a mobile robot with the pursue of studying its viability of lowering the costs as much as possible but including autonomous navigation and environment recognition capabilities.

The Project covers different subjects of engineering and robotics as it goes from the mechanical design of the robotic platform to the microcontroller software development that controls it. As the reader progresses, he will notice that the project structure follows this order so as to ease its reading.

The robot that it is presented here belongs to the educational robotics field because several Open-Source technologies had been used like Arduino and App Inventor during its design and development. The intention of the author is to share the Project as the vast majority of people that work with these same technologies do.

An innovative and original point to highlight is that the robot will be controlled by voice commands through a smartphone app specially designed and programmed to do it. This tool will allow the user to navigate the robot just by clicking one button and saying the command to the cell phone no matter where the robot is positioned in that moment. Also, he will be able to see its path and the sensors measures on the display.

Índice

Índice de figuras

Índice de tablas

Capítulo 1: Introducción

La robótica móvil esta cada vez más presente en la vida cotidiana de las personas. Actualmente existen robots comerciales muy populares que están diseñados para facilitarle la vida a los humanos por ejemplo, realizando tareas de limpieza en interiores, piscinas, placas solares y todo tipo de superficie susceptible de ser limpiada. En el ámbito domestico también existen robots que cortan el césped e incluso sillas de ruedas con navegación autónoma también se pueden considerar robots, ya que en última instancia, un robot móvil es una maquina que capaz de moverse por un entorno sin la acción directa de un ser humano.

Como es lógico pensar, las aplicaciones de la robótica móvil no están restringidas solamente al ámbito domestico. En el campo militar los robots móviles, en particular los aéreos, están provocando una revolución muy importante con la introducción de UAVs (Unmanned Aerial Vehicles) que son aeronaves capaces de realizar vuelos autónomos de larga duración. Esta tecnología también está encontrando muchas aplicaciones en la ingeniería civil para monitorizar y vigilar diferentes infraestructuras como tendido eléctrico.

En el sector aeroespacial, robots móviles terrestres son de gran utilidad para explorar las superficies de otros planetas. Un ejemplo de ellos es el robot Curiosity que actualmente se encuentra en Marte tomando muestras del planeta y analizándolas con toda la instrumentación de a bordo de la que dispone.

Sea cual sea el ámbito en el que se utilicen, el gran reto de la robótica móvil es ser capaz de implementar en los robots tecnologías que los doten de una autonomía completa para que el ser humano intervenga lo menos posible. Para alcanzar este objetivo se utilizan múltiples sensores, actuadores y procesadores de abordo en los que se implementan múltiples algoritmos de control de motores, localización, mapeo del entorno, planificación de trayectorias e incluso aprendizaje.

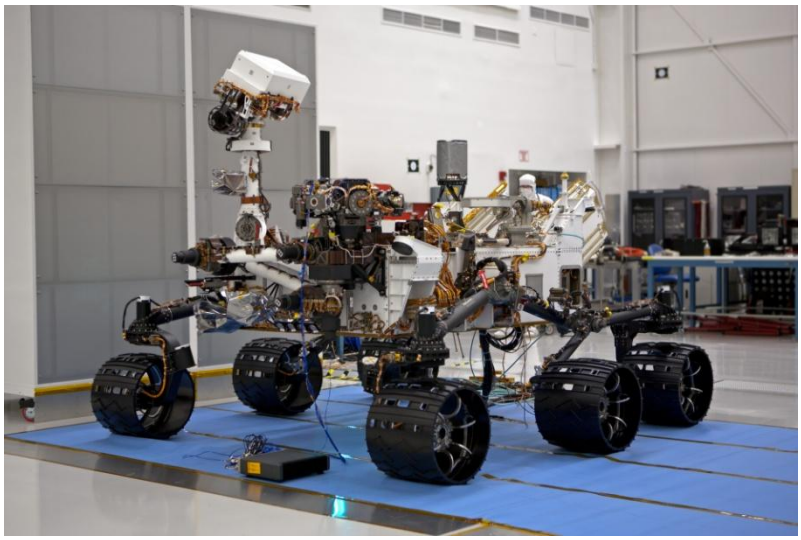


Figura 1.1 Robot Curiosity

Fuente: NASA- http://photojournal.jpl.nasa.gov/catalog/PIA14309

1.1 Objetivos

1.1.1 Objetivo general

El objetivo de este trabajo es diseñar y construir un robot móvil de bajo coste con diferentes funcionalidades de navegación autónoma y teleoperada en las que se lleve a cabo un reconocimiento del entorno por el que se mueve. Este robot debe ser controlado comandos de voz.

1.1.2 Objetivos específicos

- Diseñar y construir la plataforma robótica e integrar los diferentes elementos que la componen.
- Diseñar, programar e implementar el sistema de control y percepción del robot.
- El robot debe ser capaz de moverse por el entorno recibiendo comandos de voz a través de una aplicación de teléfono móvil utilizando tecnología Bluetooth.

- Los comandos de voz estarán definidos para que el robot reciba una orden directa de movimiento o para que el robot reciba una instrucción más compleja la realice de forma autónoma.
- Enviar la información de los sensores al teléfono móvil para que esta pueda ser monitorizada y archivada.
- Ser capaz de representar esta información en el teléfono móvil para poder visualizar un mapa del recorrido del robot donde se pueda seguir su trayectoria y los diferentes accidentes del terreno encontrados, como la inclinación y los obstáculos.

1.2 Estructura del documento

Este trabajo se divide en siete capítulos a los que se le añade al final del séptimo la bibliografía consultada y Anexos.

En el primero el lector encontrará una breve introducción al trabajo, objetivos y motivación.

El segundo capítulo corresponde al Estado del Arte donde se exponen las diferentes tecnologías y trabajos sobre los que trata este documento.

En el tercer capítulo se desarrollan las líneas generales del funcionamiento del robot así como sus requerimientos, características y arquitectura general.

El cuarto capítulo incluirá la descripción del robot y sus componentes, y los distintos procesos que se han llevado a cabo para construirlo y programarlo.

El quinto capítulo corresponde a la experimentación y pruebas de funcionamiento que se han realizado en el robot así como un análisis de los resultados y conclusiones.

En el sexto se expondrá el proceso llevado a cabo para desarrollar este proyecto así como un análisis de costes y posibles mejoras futuras.

En el séptimo y último capítulo se desarrollaran las conclusiones de este trabajo y se propondrán trabajos futuros y posibles mejoras.

Capítulo 2: Estado del Arte

En este capítulo el lector encontrará una breve introducción al mundo de la robótica móvil y las diferentes tecnologías que se están utilizando hoy en día para su desarrollo.

A medida que el lector avance, irá adquiriendo conocimientos más específicos del estado del arte sobre aspectos que están directamente relacionados con este trabajo, como Arduino, la comunicación por Bluetooth, las aplicaciones móviles en Android y la interacción robot-humano mediante comandos voz.

2.1 Robótica móvil

Actualmente, la robótica móvil es el campo donde se está desarrollando la mayor parte de la investigación académica relacionada con la robótica, en contraposición a la robótica industrial que es la que se encuentra mejor asentada en un mercado valorado en 2 mil millones de dólares [1].

El objetivo final de la mayoría de estos estudios es desarrollar mejores técnicas de control, de percepción sensorial, de localización y de navegación para implementarlas en robots móviles con el objetivo de conseguir en ellos un comportamiento completamente autónomo.

Para alcanzar esta autonomía, los sistemas de percepción sensoriales del robot deben ser capaces de conocer su posición respecto en relación a su entorno mediante la creación de un mapa. Utilizando técnicas de SLAM (Simultaneous Location And Mapping) el robot puede localizarse y construir el mapa a medida que se mueve [2].

Conociendo el entorno y su posición, las diferentes técnicas de planificación de trayectorias permiten al robot calcular el camino que ha de seguir para alcanzar su objetivo de una manera eficiente. Este conjunto de técnicas producen como resultado una trayectoria a seguir que servirá de referencia para el sistema de control de los motores desarrollando las velocidades precisas en cada instante para realizarla [3].

2.1.1 Robótica educacional y recreativa

Un aspecto muy característico de la robótica en general es que asombra a la gente, por eso muchas de estas personas deciden investigar y participar en este mundo de forma aficionada o incluso profesional.

Competiciones de lucha de robots [4], carreras de robots que "siguelíneas"[5] o incluso partidos de futbol entre robots [6] no son más que otras de las muchas actividades que existen hoy en día en relación a la robótica móvil y que tanto se están extendiendo.

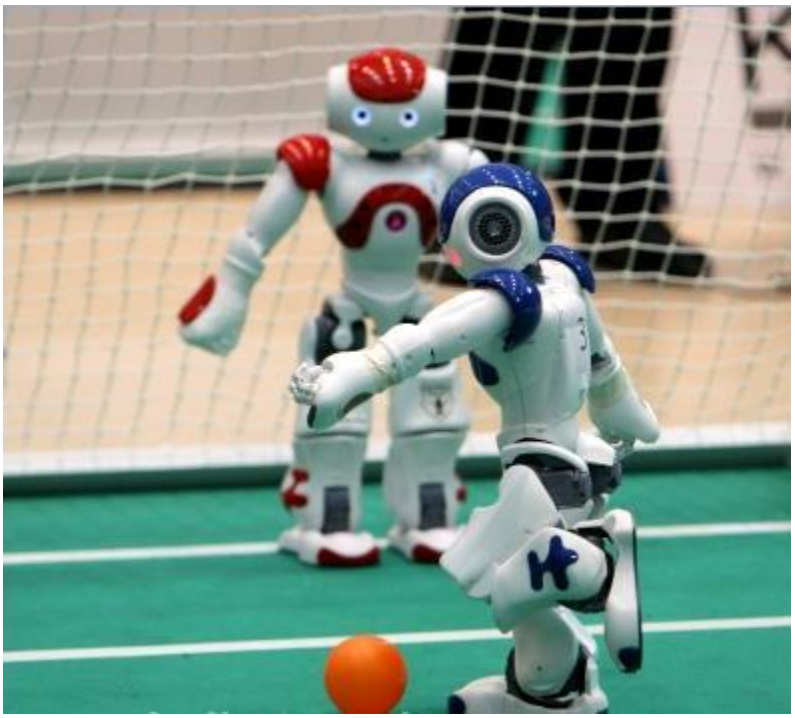


Figura 2.1 Robot antropomórfico jugando al fútbol
Fuente: Robocup-http://www.robocup2014.com

Puede que la robótica educacional en algunos aspectos no esté a la vanguardia de la técnica ni de la investigación académica pero tiene unan gran virtud, y es la cultura en compartir y difundir conocimientos que le rodea, en la que busca introducir a todo el mundo, sobre todo a las futuras generaciones [7].

2.2 Arduino

Arduino es una placa de desarrollo electrónica basada en el microcontrolador ATmega328 [8].

A primera vista, puede parecer un microcontrolador estándar, pero las intenciones de sus creadores han sido las que han provocado una revolución en el mercado y lo ha hecho tan famoso. Y es que, tanto el microcontrolador como su entorno de desarrollo han sido diseñados para acercar a todo el mundo al prototipado de sistemas electrónicos en diferentes campos.

Debido a su filosofía Open-Source, Arduino se ha convertido en la plataforma hardware más famosa de la actualidad [9] [10] [11] [12], lo que ha favorecido que los diferentes desarrolladores hayan creado una comunidad muy extensa en la que comparten su software con el resto del mundo, difundiendo muy rápidamente el conocimiento gracias a páginas web como GitHub o la propia de Arduino.

Figura 2.2 Arduino UNO
Fuente: Arduino- http://www.arduino.cc

A día de hoy, Arduino ha creado su propio nicho en la industria de la electrónica, en la que existen alrededor del mundo más de 200 distribuidores de sus productos incluyendo grandes compañías como Sparkfun Electronics en Estados Unidos.

2.2.1 Arduino en la robótica móvil

El mayor impacto que ha tenido Arduino en la robótica móvil sin duda ha sido en el ámbito educacional y académico, en el que ha resultado ser una herramienta muy poderosa para introducir a los estudiantes a esta tecnología [13] [14] [15].

Arduino se ha utilizado para controlar desde los motores un robot [16] hasta para leer y procesar datos de múltiples sensores y realizar una conexión inalámbrica con un PC para enviar esta información [17].

Al ser programable en lenguaje C, facilita mucho el desarrollo de software dada la popularidad de este lenguaje.

Un ejemplo de robot móvil es Boe-Bot de la compañía Paralax [18]. En este robot, la Universidad Eslovaca de Tecnología en Bratislava ha sustituido el microcontrolador que lleva incorporado de serie por un Arduino UNO. El nuevo robot ha sido bautizado como Acrob [14] y está orientado a la educación en robótica móvil y programación de microcontroladores.

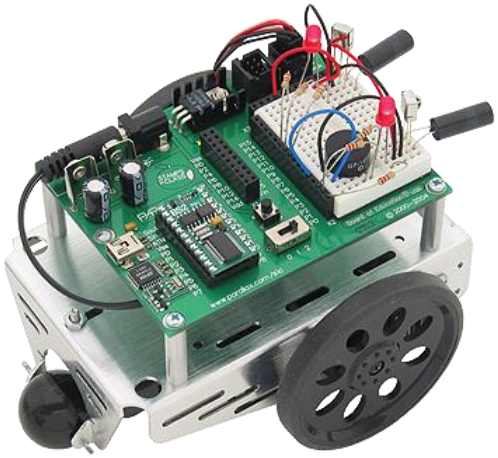


Figura 2.3 Robot Boe-Bot
Fuente: Paralax-http://www.paralax.com

La propia empresa que comercializa Arduino también ha creado una plataforma robótica denominada Arduino Robot [8] en la que vienen integrados dos microcontroladores con una serie de sensores infrarrojos, motores y baterías.

Siguiendo su misma filosofía, ofrecen la plataforma integral que puede ser programada igual que el resto de sus placas. Esto favorece que el proceso de desarrollo y aprendizaje se centre más en la parte de software que en la de hardware que ya viene dada y no permite muchas modificaciones.

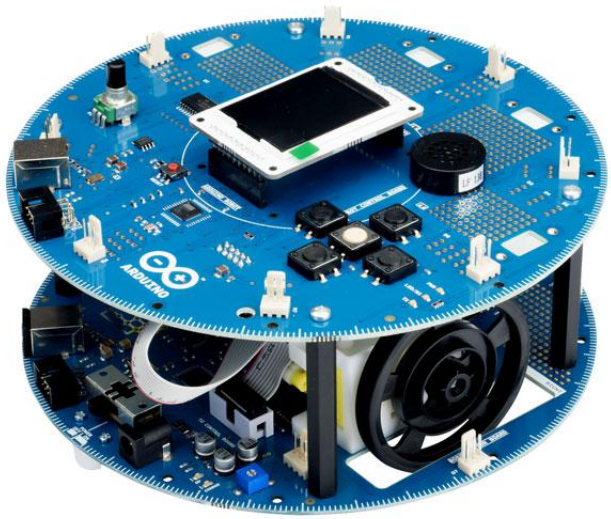


Figura 2.4 Arduino Robot
Fuente: Arduino-http://www.arduino.cc

2.3 Comunicación Bluetooth

La comunicación Bluetooth es un protocolo orientado a la conectividad inalámbrica entre una gran variedad de dispositivos electrónicos.

Está basada en un enlace de radiofrecuencia de baja potencia, pero proporciona conexiones instantáneas entre los dispositivos sin necesidad de utilizar un punto de acceso intermedio como un router.

Debido a que Bluetooth es un estándar de comunicaciones abierto y global es posible conectar distintos dispositivos de diferentes fabricantes lo que facilita una conexión versátil y de bajo coste.

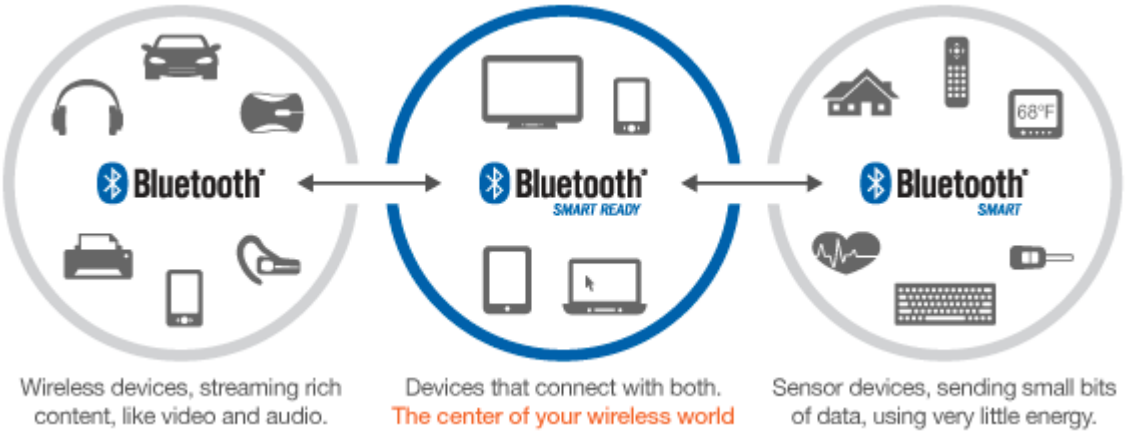


Figura 2.5 Dispositivos con conexión Bluetooth

Fuente: Bluetooth-http://www.bluetooth.com

En la actualidad, aparte de su importancia en la comunicación de ratones, teclados, mandos de videoconsolas e impresoras, esta tecnología está experimentando un gran auge en la electrónica de consumo, en productos como auriculares, altavoces y manos libres, ya que al estar implementada en cualquier teléfono móvil, permite controlar estos dispositivos de manera inalámbrica aportando un gran valor añadido al producto [19] [20].

2.3.1 Comunicación Bluetooth entre un robot y un móvil

Durante los últimos años la tendencia ha sido la comunicación entre el robot y un PC como puede ser el robot YaMoR [21]. Este robot incorpora un módulo de Bluetooth que actúa de interfaz entre el puerto serie de su microcontrolador y el PC.

Sin embargo, gracias a la gran capacidad de cómputo de los teléfonos móviles actuales, su portabilidad y versatilidad, hace que estos sean cada vez más una plataforma idónea para llevar a cabo las tareas que puede hacer el PC [22].

Establecer una conexión mediante Bluetooth entre un móvil y un microcontrolador abre al usuario muchas posibilidades que pueda controlar su robot y monitorizar su estado [23] [24].

2.4 Aplicaciones móviles en Android.

Desde el punto de vista del teléfono móvil, el desarrollo de aplicaciones que facilitan y automatizan la conexión mediante Bluetooth y el tratamiento de datos es imprescindible [25].

Android ofrece un sistema operativo muy accesible para los desarrolladores e investigadores y permite crear aplicaciones específicas para cada caso. El lenguaje utilizado para programar estas aplicaciones es Java, pero existen muchas plataformas para su desarrollo que no requieren conocimiento de este lenguaje, una de ellas es App Inventor.

En el ámbito de la robótica educacional, los trabajos de Roca Soler, J. [26] o el de Bake Fajardo, M. G. y Moreno Godoy, H. G. [27], son dos de los muchos que existen sobre la comunicación de un terminal Android con un robot usando una aplicación móvil.

2.4.1 App Inventor

App Inventor es una herramienta web gratuita desarrollada por el profesor Hal Abelson del MIT junto con un equipo de Google Education [28].

Esta herramienta permite una programación basada en la conexión de diferentes bloques lógicos, facilitando la programación de aplicaciones para teléfonos Android sin necesidad de conocer el lenguaje Java.

Es por esto que ha supuesto un importante avance en el ámbito educativo, permitiendo a muchos estudiantes desarrollar sus primeras aplicaciones de

manera sencilla al encontrarse una herramienta con una curva de aprendizaje muy reducida, permitiendo potenciar rápidamente su creatividad [29].

App Inventor da soporte a una comunidad de casi dos millones de usuarios de 195 países de todo el mundo y gracias a ella se han desarrollado 4,7 millones de aplicaciones hasta la fecha. En su deseo de promover el aprendizaje de la programación, ha captado la atención de varios sectores de la población como educadores, investigadores, emprendedores y aficionados a la tecnología.

En el ámbito de la robótica, se han desarrollado algunos trabajos utilizando esta plataforma, por ejemplo para monitorizar un sistema goniométrico que permite calcular la posición y orientación de un robot [30], o para teleoperar directamente un robot mediante Bluetooth y monitorizar ciertas variables de sus sensores, como la aplicación Sparki Bluetooth Controller [31], adaptada para el robot Sparki de la compañía Arcbotics [32].

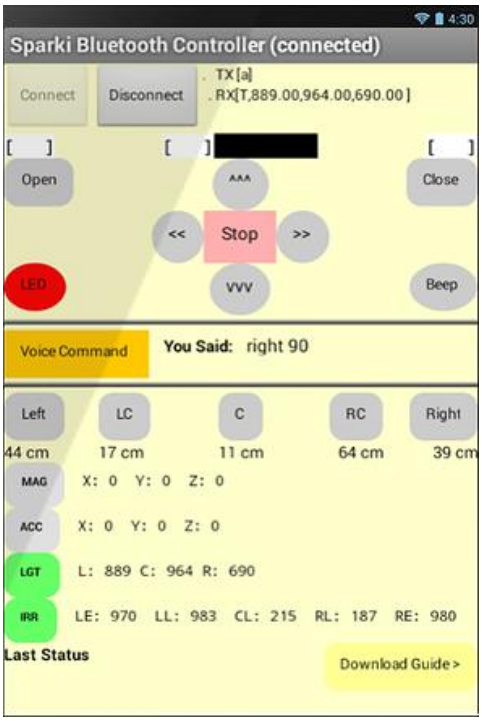


Figura 2.6 Aplicación de App Inventor para el robot Sparki

Fuente: SJDavis Robotics-http://www.sjdavisrobotics.com

2.5 Interacción Humano-Robot

La interacción humano robot o sus siglas en ingles HRI (Human Robot Interaction) es un campo novedoso en el mundo de la robótica en el que el conjunto de investigadores que participan se caracteriza por tener un perfil multidisciplinar en el que trabajan tanto sociólogos, ingenieros y educadores entre otras ramas.

La interacción humano robot tiene como objetivo el estudio y el desarrollo de diferentes tecnologías que faciliten la interacción natural entre un humano y un robot, como pueden ser la compresión y el habla del lenguaje, el reconocimiento de voz y la interpretación de gestos y expresiones.

Dada la evidente amplitud que caracteriza a este campo, en el siguiente apartado se expondrá el estado del arte de la tecnología STT (Speech To Text) que es utilizada en este trabajo y cuya finalidad es transformar órdenes de voz a texto digital para que pueda interpretarlo un computador [33].

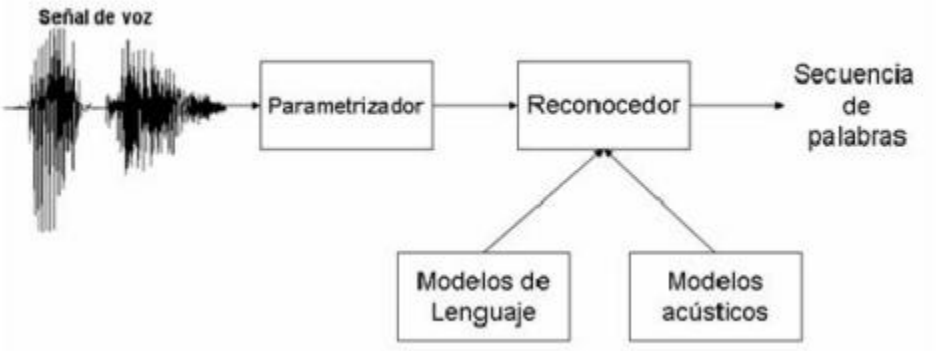


Figura 2.7 Diagrama de un sistema de reconocimiento de voz

2.5.1 Comandos de voz mediante conversión de voz a texto

El uso de comandos de voz para controlar sistemas es una práctica que facilita mucho su manejo al eliminar interfaces graficas y táctiles como un mando o un panel de control.

Gracias a ello se ha podido utilizar en aplicaciones donde las personas tienen algún tipo de deficiencia motriz [34] permitiéndoles controlar diferentes aparatos domésticos con comandos de voz.

En el ámbito de la robótica también se utiliza para controlar robots móviles educacionales [35] o sillas de ruedas para personas con dificultades motrices [36].

Los trabajos citados anteriormente utilizan un modulo electrónico de reconocimiento de voz denominado EasyVR de la compañía VeaaR [37]. Este modulo permite pre grabar ciertas ordenes y posteriormente captar por un micrófono que lleva incorporado, los comandos de voz recibidos y compararlos con los guardados.

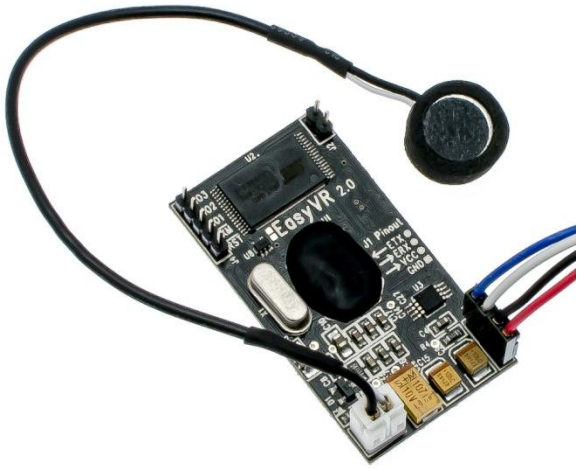


Figura 2.8 Modulo EasyVR 2.0
Fuente: VeeaR -http://www.veear.com

Capitulo 3: Requisitos del sistema

En este capítulo se describirán las diferentes funcionalidades deseadas del robot así como sus requisitos y su arquitectura necesaria.

3.1 Sistema de locomoción

El robot que se va a construir es un robot de tipo diferencial.

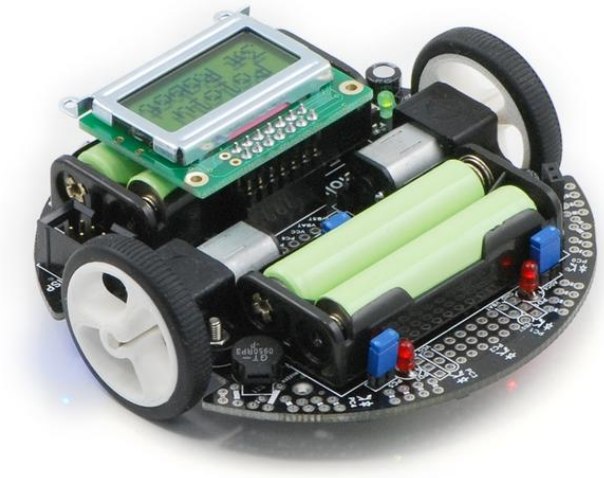


Figura 3.1 Locomoción diferencial en el robot Pololu
Fuente: http://www.pololu.com

Los motivos de esta decisión son los siguientes:

- La mayor flexibilidad y maniobrabilidad a la hora de moverse por terrenos angostos ya que puede rotar sobre su propio eje.
- La facilidad de control al no tener un sistema de dirección ajeno a las ruedas motrices como el sistema Ackerman [1].
- El bajo coste de piezas y componentes al precisar solo dos ruedas motrices

Debido a que un robot diferencial solo tiene dos ruedas motrices, es necesario un punto de apoyo extra para conseguir estabilidad. En los robots diferenciales suele montarse una rueda omnidireccional que no es motriz pero permite el movimiento en cualquier dirección.

Figura 3.2 Rueda omnidireccional
Fuente: http://www.pololu.com

En este robot es obligatorio incluir solo una rueda loca en la parte trasera debido a que se requiere que el robot se mueva por terrenos con desnivel y si existieran más de tres puntos de apoyo sería necesario un sistema de amortiguación para que todas las ruedas tomaran contacto con el suelo ante un cambio de pendiente [1].

3.2 Estudio cinemático

El estudio cinemático de un robot móvil es necesario si se quiere conocer como se mueve y como cambia su posición según sus movimientos.

En este apartado se desarrolla un estudio de la cinemática de un robot diferencial genérico.

En la Figura 3.3 se pueden observar los parámetros físicos característicos de un robot diferencial. Estos son: el radio del robot, b; el radio de la rueda, r; la velocidad del robot, V; la velocidad de giro del robot, $\dot{\varphi}$; y la velocidad de giro de las ruedas, $\omega 1$ y $\omega 2$.

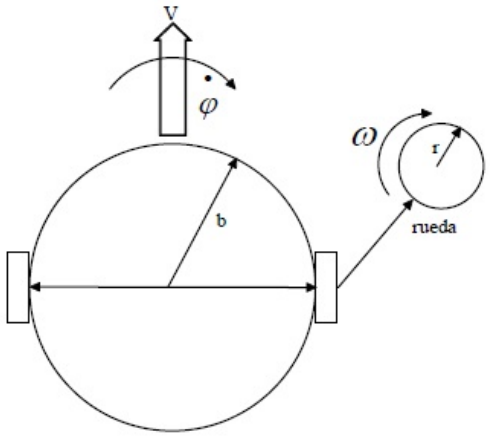


Figura 3.3 Parámetros cinemáticos de un robot con tracción diferencial
Fuente: González, L. H. R., López, M. B. (2008) Modelo matemático para un robot móvil.

3.2.1 Ecuaciones cinemáticas

Las ecuaciones cinemáticas que se quieren hallar relacionan la velocidad de giro de las ruedas con las variables que definen la postura del robot (Figura 3.4).

Estas variables son: coordenada x del centro geométrico del robot, coordenada y del centro geométrico del robot y el ángulo φ que forma el eje Y del sistema de referencia móvil con el eje X_0 del sistema de referencia fijo.

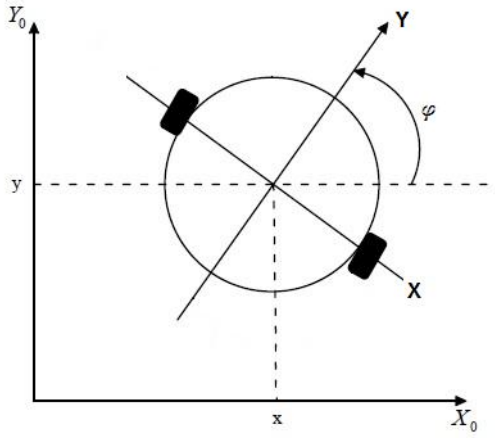


Figura 3.4 Postura de un robot

La velocidad lineal V se obtiene a partir del promedio de las velocidades lineales de cada una de las ruedas estas se obtienen como el producto de su velocidad angular $w1$ y $w2$ por el radio r. La velocidad lineal queda definida por:

$$V = \frac{r \cdot (w1 + w2)}{2} \qquad [3.1]$$

El ángulo de giro del robot φ queda determinado a partir de las relación geométrica entre el movimiento de cada una de las ruedas del robot, tal como se puede ver en la siguiente figura.

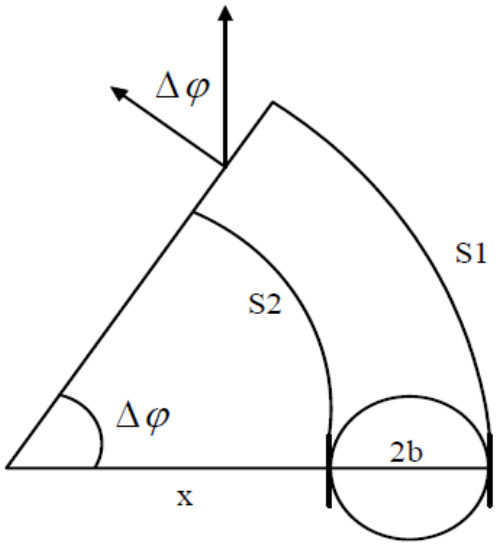


Figura 3.5 Cinemática de un robot con tracción diferencial
Fuente: Luis Hernando Ríos G. Maximiliano Bueno L. Modelo matemático para un robot móvil

El ángulo girado por el robot es el ángulo del arco contenido por la trayectoria. Conociendo que la rueda derecha gira a mayor velocidad que la izquierda, el ángulo de dirección debe aumentar $\Delta\varphi$. Según se puede ver en la Figura 3.5, la rueda izquierda traza un arco de radio x.

La distancia recorrida por esa rueda será:

$$S2 = \Delta\varphi \cdot x \quad [3.2]$$

Como la rueda izquierda se encuentra más alejada del centro de giro, esta recorre una distancia mayor en el mismo tiempo, dada por:

$$S1 = \Delta\varphi \cdot (x + 2b) \quad [3.3]$$

Restando S2 a S1 se puede despejar fácilmente $\Delta\varphi$, cuya expresión es la siguiente:

$$\Delta\varphi = \frac{S1 - S2}{2b} \quad [3.4]$$

Al dividir por el tiempo transcurrido, se obtiene la velocidad de giro del robot en función de la velocidad de cada una de sus ruedas, como se indica en la ecuación:

$$\dot{\varphi} = \lim_{\Delta t \to 0} \frac{\Delta\varphi}{\Delta t} = \frac{r \cdot (w1 - w2)}{2b} \quad [3.5]$$

Para obtener la postura del robot hay que descomponer la velocidad lineal del robot en las velocidades asociadas a cada eje del plano XY. Esto se expresa a partir de las ecuaciones 3.6 y 3.7.

$$\dot{X} = V \cdot \cos(\varphi) \quad [3.6]$$

$$\dot{Y} = V \cdot \sin(\varphi) \quad [3.7]$$

3.3 Funcionalidades del robot

En este apartado se describirán las funcionalidades deseadas del robot así como ciertos requisitos que ha de cumplir.

3.3.1 Capacidades motrices

El robot debe ser capaz de moverse en todas las direcciones del espacio. Los movimientos que se van a implementar son de avance y rotación. Esto permite que el robot se oriente en la dirección deseada y que se traslade de forma rectilínea.

Para realizar un movimiento en línea recta, ambos motores deben girar a la misma velocidad ya que si no fuera así se descompensaría la trayectoria.

Para ello es necesario diseñar un sistema de control en lazo cerrado que regule sus velocidades. Esto se consigue implementando un sistema de encoders que obtienen estos valores para que un microcontrolador los procese.

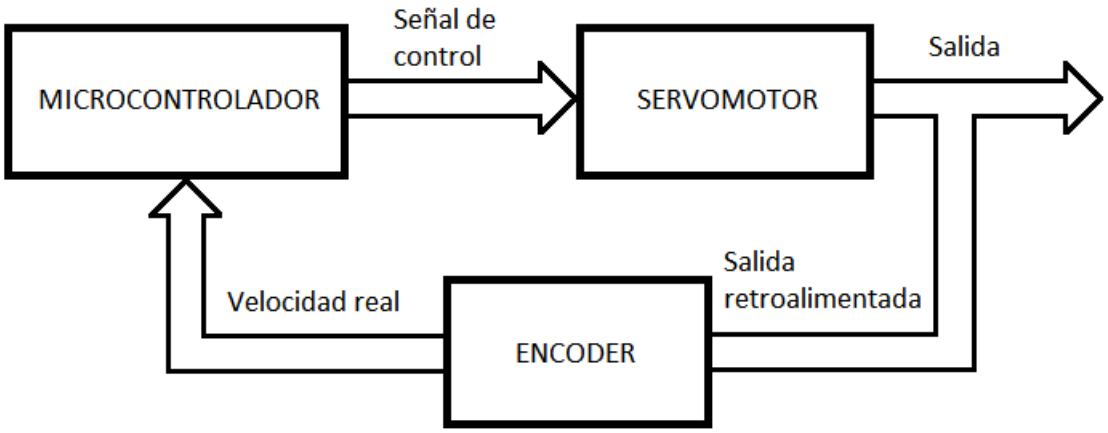


Figura 3.6 Sistema de control en lazo cerrado

Otra funcionalidad es que el robot debe ser capaz de girar un cierto número de grados exactos dependiendo de la orden recibida. Para ello se utiliza ese mismo sistema de encoders.

3.3.2 Algoritmos de navegación

El robot debe ser capaz de detectar y evitar obstáculos, seguir paredes y regresar a su posición inicial desde cualquier punto del espacio.

Para realizar estas diferentes acciones se debe diseñar un sistema de navegación en el que se implementen sensores exteroceptivos que recopilen la información del exterior y sensores propioceptivos (encoders) que permitan al robot conocer su posición en cada momento.

Los sensores exteroceptivos deben ser controlados por el microcontrolador y son necesarios dos, uno en la parte frontal del robot y otro en la parte lateral que evalúen las distancias a la que se encuentran de los obstáculos.

Los algoritmos de navegación que se van a implementar son los siguientes.

Parada automática

El robot debe ser capaz de pararse automáticamente cuando se dirige hacia un obstáculo y su distancia sea menor que un valor definido.

Evasión de obstáculos

El robot debe ser capaz de esquivar obstáculos independientemente de su forma. Una vez esquivado, el robot debe seguir la misma trayectoria que seguía antes de esquivarlo.

Seguimiento de paredes

El robot debe ser capaz de seguir paredes independientemente de su forma. Esta funcionalidad tiene que permitir realizar un reconocimiento de habitaciones, la navegación en caminos con paredes laterales y la resolución de laberintos.

3.3.3 Reconocimiento de las irregularidades del terreno

Dadas las capacidades de exploración y reconocimiento que se desean caracterizar en el robot, se requiere que reconozca las irregularidades del terreno.

Para ello se incluye un sensor que mide las diferentes aceleraciones a las que se expone para posteriormente calcular las inclinaciones del robot, tanto frontales como laterales.

Estas medidas aportarán una buena noción de cómo es el terreno por el que se está desplazando el robot.

3.3.4 Interacción humano-robot

La manera de la que se desea que el usuario pueda interactuar y controlar el robot es mediante comandos de voz.

Estos son ordenes de movimiento directo (avanzar, girar 45 grados a la izquierda, dar media vuelta, etc.) o de movimiento autónomo (esquivar el obstáculo y seguir las paredes).

Al principio de este proyecto se adquirió una placa EasyVR a la que se ha hecho referencia en el punto 2.5.1 pero no se llegó a implementar porque no reconocía los comandos si el robot se alejaba mucho y por lo tanto no cumplía los requisitos.

Por ello que se decidió utilizar un teléfono móvil que permaneciera siempre con el usuario y además permitiera visualizar información del robot por pantalla en tiempo real.

El teléfono móvil que se va a utilizar esta basado en el sistema operativo de Android y actuará de interfaz entre el usuario y el robot a través de una red de comunicación inalámbrica.

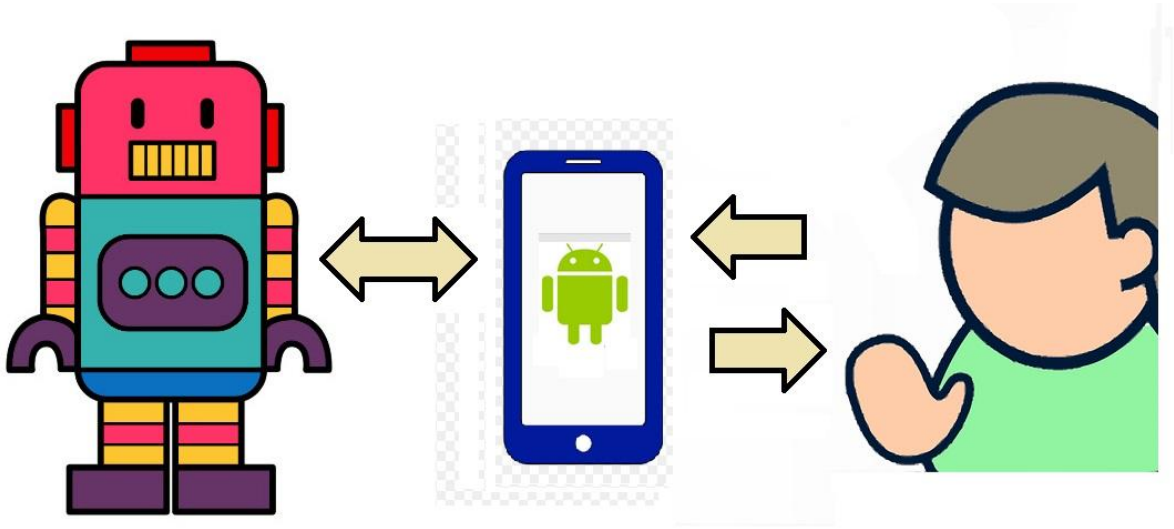


Figura 3.6 Interacción humano propuesta robot a través del teléfono móvil

Existen dos maneras fáciles y económicas de establecer esta comunicación, y son el Bluetooth o el WiFi pero para este proyecto se ha decidido utilizar comunicación Bluetooth por su facilidad de uso y menor coste y consumo en comparación con la comunicación WiFi.

3.4 Aplicación móvil

Para facilitar el proceso de comunicación se diseñará una aplicación móvil que automatice los siguientes procesos.

1. El establecimiento de la conexión.
2. Interpretación de los comandos de voz.
3. Envío de las correspondientes instrucciones al robot.
4. Recibimiento y representación del valor de sus sensores.
5. Representación en un mapa de su posición, trayectoria y obstáculos que se detectan.
6. Almacenamiento de todos los datos.

3.5 Arquitectura del sistema

Basándose en los requerimientos que se han ido exponiendo a lo largo de este capítulo, a continuación se mostrará la arquitectura general del sistema que permitirá cumplir estos objetivos y funcionalidades.

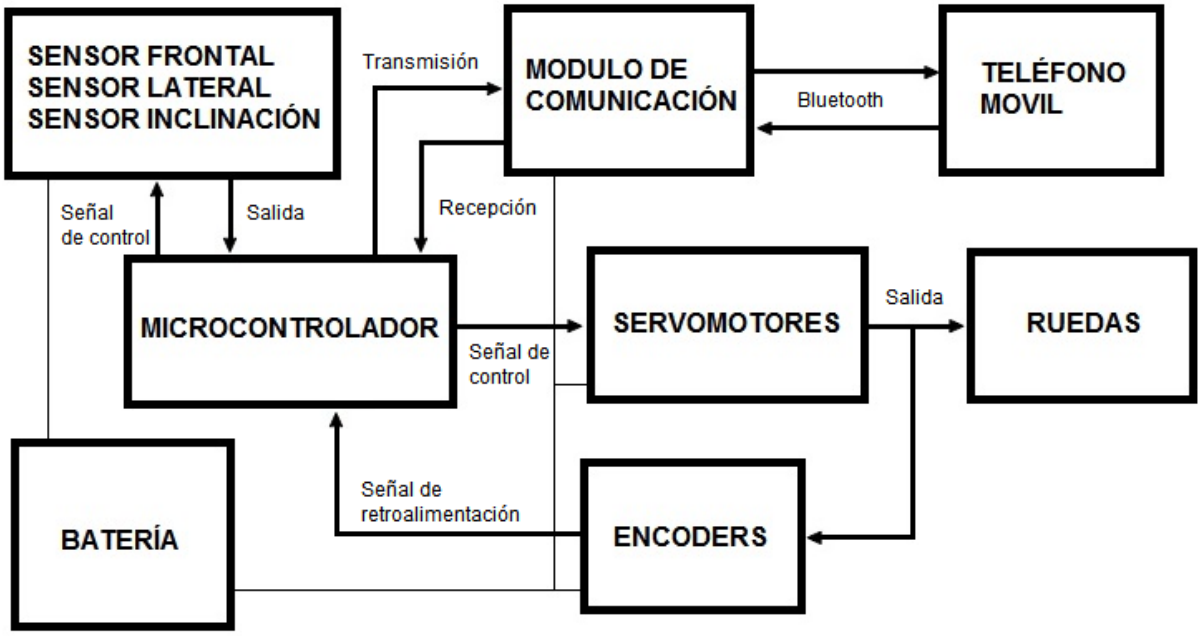


Figura 3.7 Arquitectura del sistema

Capítulo 4: Descripción del sistema

En este capítulo se describirá todo el proceso seguido para el diseño y fabricación del robot.

Se seguirá un desarrollo bottom-up, donde se empezará por el diseño mecánico de la estructura del robot pasando por la sensorización, localización y programación del microcontrolador hasta finalizar con el desarrollo de la aplicación móvil.

4.1 Diseño mecánico y construcción de la plataforma

Aunque en este apartado todavía no serán descritos los componentes que forman el robot, se han tenido en cuenta a la hora de diseñar su estructura ya que las medidas de estos componentes condicionan las medidas de esta. Es por eso que se ha decidido construir un modelo 3D del cuerpo del robot y de sus componentes más relevantes para poder modificar las medidas, en función de las necesidades, hasta alcanzar un diseño final que posteriormente ha sido fabricado.

El modelo 3D se ha realizado con el software SolidWorks de la compañía SolidWorks Corp. Este software es un programa de diseño asistido por ordenador o más conocido por sus siglas en ingles CAD (Computer-Aided Desing) que permite el modelado de sólidos de manera paramétrica.

En este trabajo no se va a proporcionar una descripción exhaustiva del proceso de diseño del modelo 3D. Se proporcionara el diseño final y los planos acotados (Anexos) e imágenes en 3D de las diferentes partes del robot indicando los componentes que sustentan y como se fijan entre ellos.

4.1.1 Base

La plataforma del robot es la pieza donde se sustentará toda la estructura del robot y donde irán fijados los servomotores, las ruedas y la batería.

Se han tenido muy en cuenta las medidas de estos componentes ya que condicionaban mucho las dimensiones de esta base.

En las siguientes figuras se mostrara una vista en isométrico, planta y alzado de la plataforma. Los planos acotados se incluyen en los Anexos.

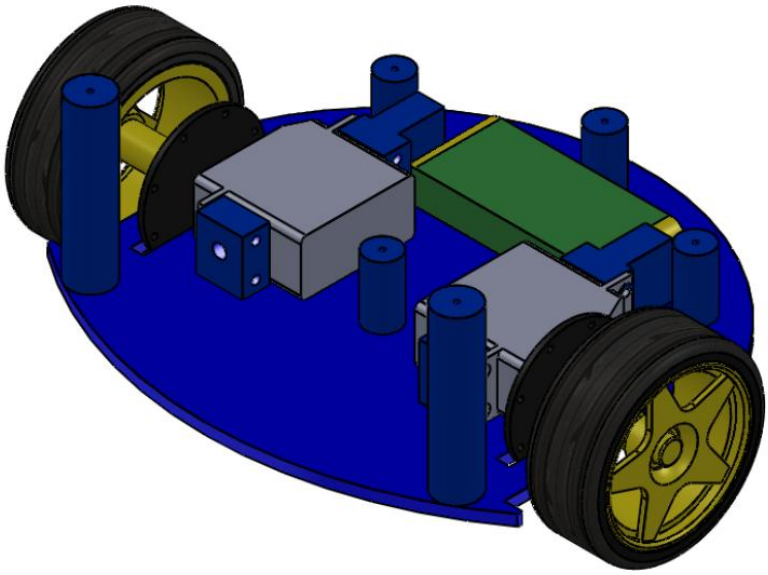


Figura 4.1 Vista isométrica de la base

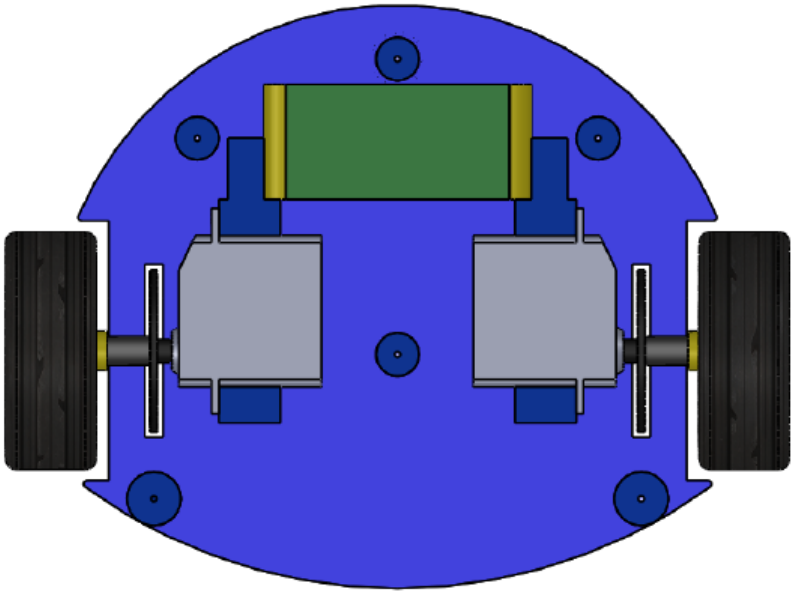


Figura 4.2 Planta de la base

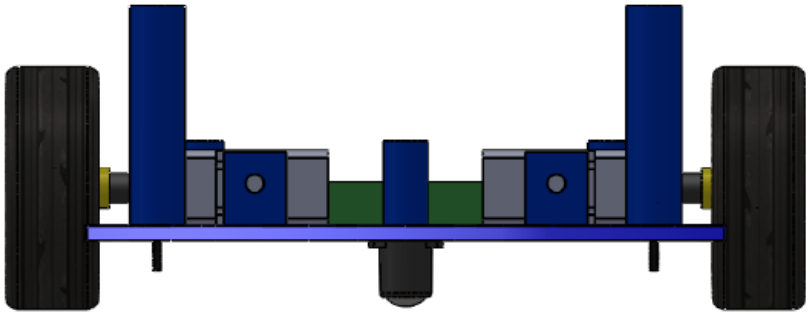


Figura 4.3 Alzado de la base

4.1.2 Plataforma intermedia

La plataforma intermedia es la pieza donde se sustenta el microcontrolador, la antena de Bluetooth y el acelerómetro. Esta pieza está fijada a las columnas de la base con unos tornillos para madera con métrica de 2 milímetros.

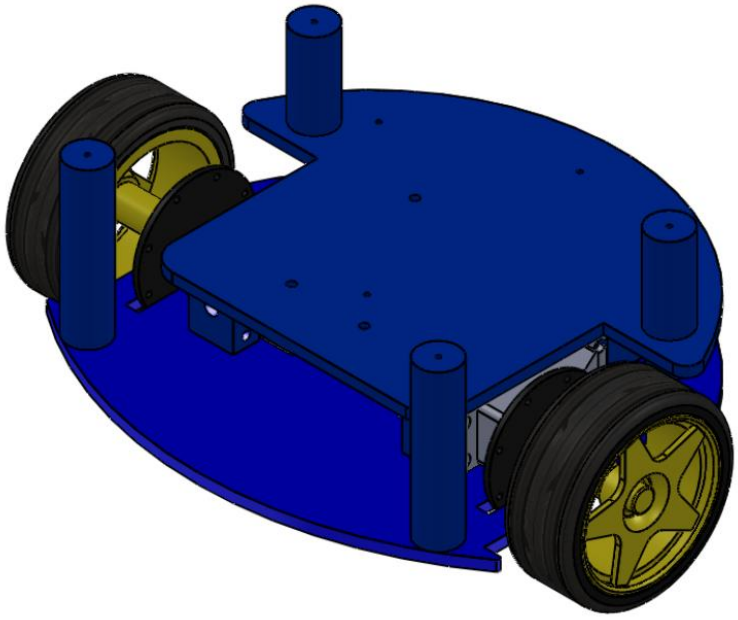


Figura 4.4 Vista isométrica de la plataforma intermedia

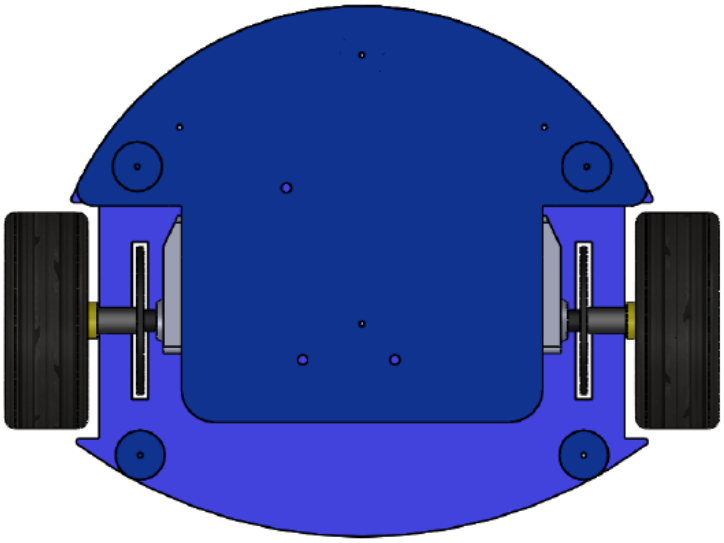


Figura 4.5 Planta de la plataforma intermedia

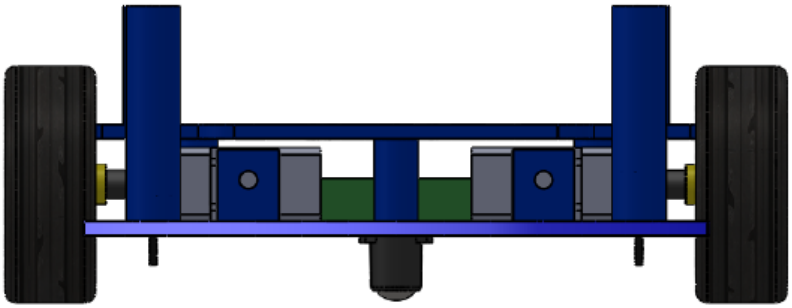


Figura 4.6 Vista isométrica de la base del robot

4.1.3 Cubierta y parachoques frontal

La cubierta del robot es la pieza que cubre el robot por arriba y se encarga de cubrir los componentes y el cableado. Está fijada a la plataforma intermedia y a la base con tornillos para madera de con métrica de 2 milímetros. Finalmente, el parachoques frontal va adherido a la base del robot.

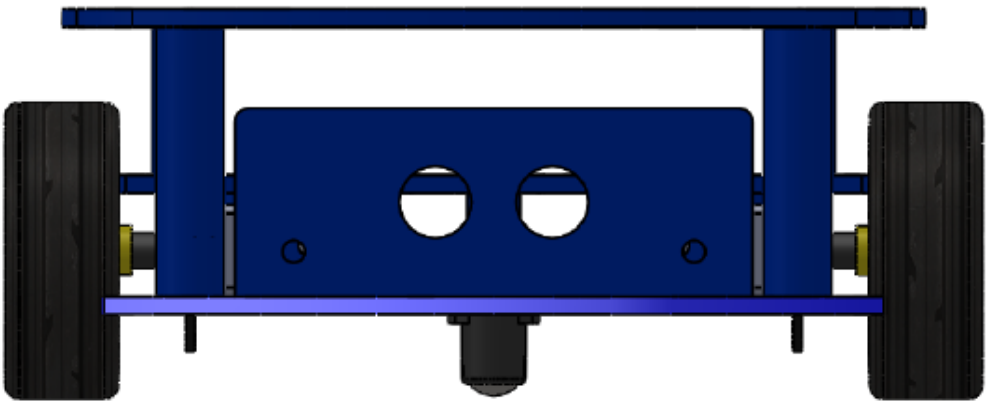


Figura 4.7 Alzado del robot completo

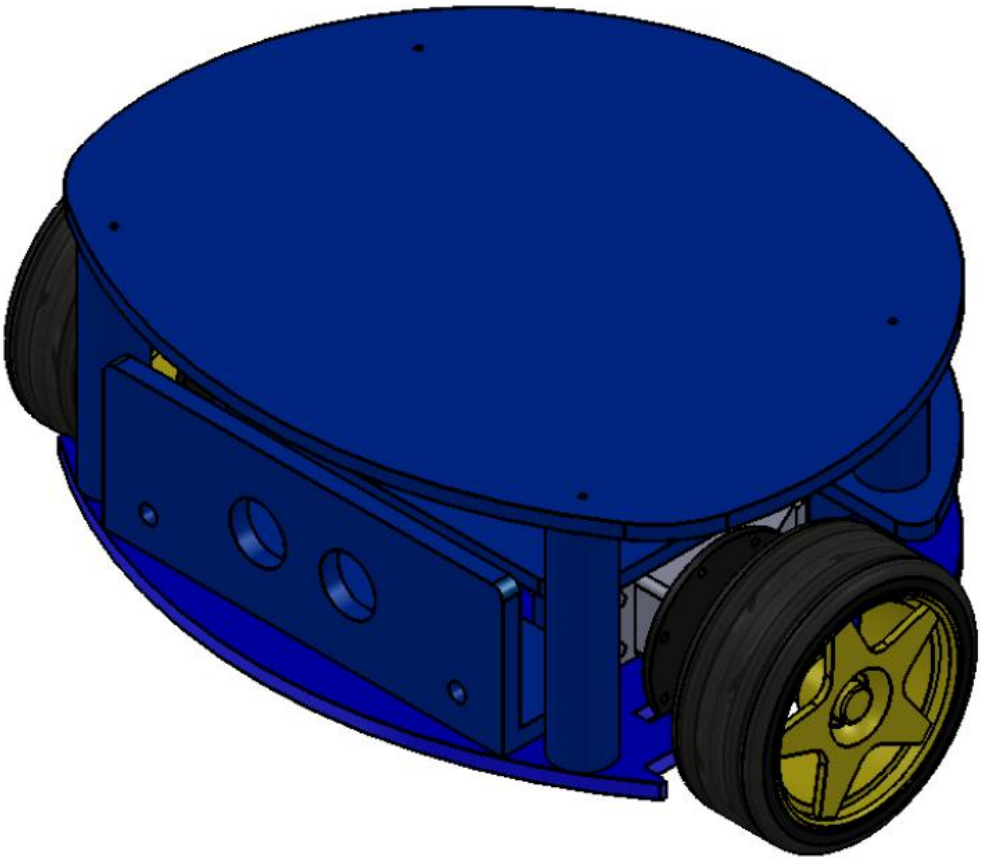


Figura 4.8 Vista isométrica del robot completo

4.1.4 Fabricación

Para la fabricación del robot se ha utilizado madera de contrachapado de pino con un grosor de 60 milímetros. Se ha decidido utilizar este material porque es ligero, barato y fácilmente moldeable.

A parte, se han utilizado 8 tornillos para madera de métrica 2 milímetros y de longitud 15 milímetros, y 4 tornillos pasantes con tuerca de métrica 3 milímetros y longitud de 25 milímetros.

Para realizar las piezas del robot se han utilizado las una sierra de calar con hoja para madera, un taladro, una lijadora eléctrica, cola adhesiva y un spray de pintura azul.

El proceso de fabricación de cada pieza siguió los siguientes pasos. Primero se trasladaron los planos en 2D del modelo a la pieza para posteriormente ser recortados con la sierra de calar.

Seguidamente se cortaron las piezas de apoyo entre plataformas y se realizaron los taladros necesarios para los tornillos.

Después se lijaron todas las piezas y se pegaron utilizando la cola adhesiva. Finalmente se pintaron, se dejaron secar y una vez secas se montaron todos los componentes y se enroscaron los tornillos.

.

Figura 4.9 Robot completo

4.2 Descripción del Hardware

En este apartado se describirá el microcontrolador Arduino UNO, sensores, actuadores y circuitos de alimentación que conforman el robot. Se expondrá el motivo por el cual se utilizará cada elemento, una breve explicación del principio de funcionamiento, el método de implementación utilizado y posibles cálculos y graficas que describan su comportamiento.

4.2.1 Arduino UNO

Arduino UNO llega incorporado un microcontrolador ATmega 328. Este es un microcontrolador de 8 bits y de bajo consumo del fabricante Atmel.

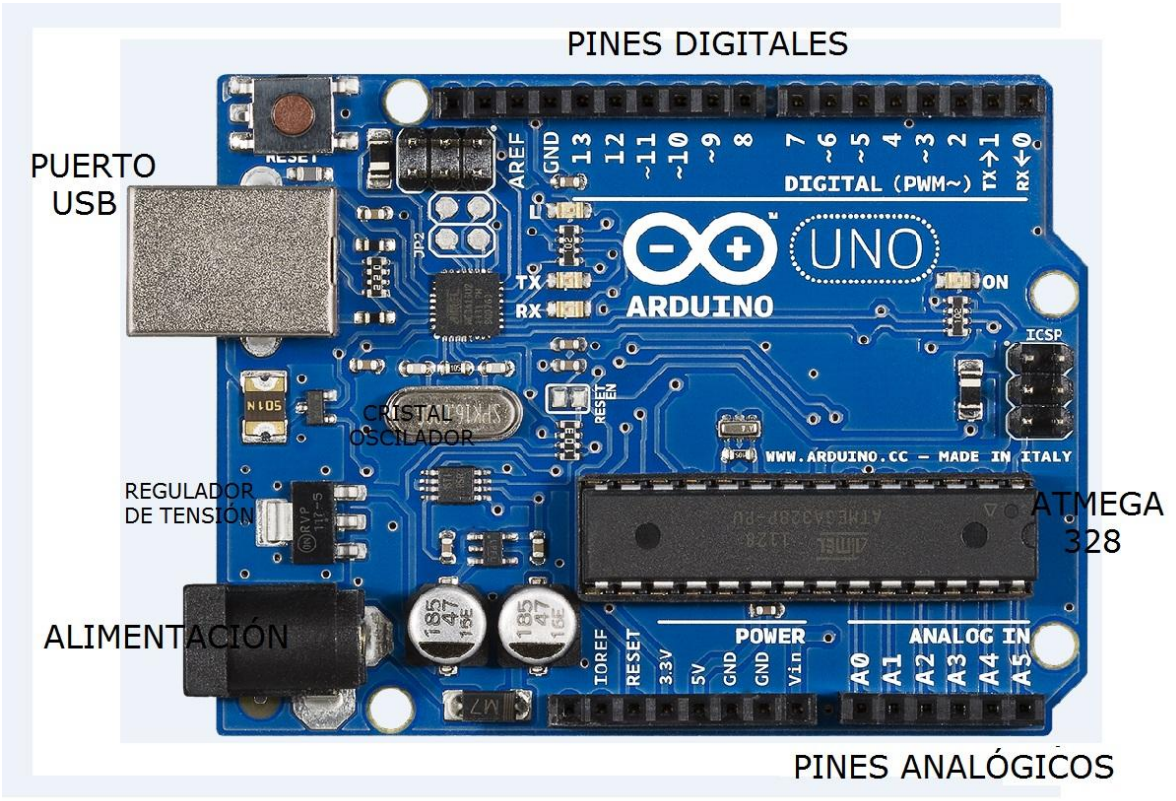


Figura 4.12 Componentes de un Arduino UNO

ATmega328

El ATmega328 está basado en una arquitectura AVR cuyo procesador permite ejecutar 131 tipos de instrucciones diferentes, casi todas en un solo ciclo de reloj y está compuesto por 32 registros de carácter general conectados directamente a la ALU (Arithmetic-Logic Unit).

Memoria

La memoria de código del ATmega328 es una memoria tipo Flash de 32Kb.

En esta memoria se almacenan las instrucciones de programa y estas pueden borrarse y reescribirse de manera eléctrica mediante un puerto USB (Universal Serial Bus) que se conecta al ordenador y permite al usuario modificar el programa como lo desee.

La memoria de datos del microcontrolador es de tipo SRAM (Static Random Access Memory) con una capacidad de 2Kb.

Convertidor ADC

El microcontrolador ATmega328 posee diferentes periféricos, entre ellos un convertidor analógico digital de 10 bits y de aproximaciones sucesivas SAR (Sucesive Approximation Register).

Por defecto, la conversión se hace de 0 a 5 voltios pero utilizando el pin ARef se puede modificar la tensión de referencia del convertidor para incrementar su límite superior.

Interrupciones

El microcontrolador es capaz de atender 24 tipos de interrupciones incluyendo interrupciones externas en los pines 2 y 3 e interrupciones internas generadas por los timers y por los protocolos de comunicación.

Comunicaciones

Arduino UNO puede comunicarse con otros dispositivos de diferentes formas. El ATmega328 proporciona un modulo USART (Universal Synchronous and Asynchronous serial Reciver and Transmitter) en los pines 0 (RX) y 1(TX).

También incluye un puerto USB (Universal Serial Bus) para comunicarse con un ordenador y así poder cargar el software que se requiere programar en el microcontrolador.

ATmega328 también admite comunicación SPI (Serial Peripherial Interface) e I²C (Inter-Integrated Circuit) que permiten una comunicación síncrona de alta velocidad con otros dispositivos.

Alimentación

Arduino UNO puede alimentarse mediante el puerto USB a 5 voltios o puede conectarse una fuente externa de alimentación como una batería utilizando el conector Jack que incluye. La fuente de alimentación externa puede tener una tensión entre los 6 y los 20 voltios aunque el rango recomendado es entre 7 y 12 voltios.

La placa tiene unos pines que proporcionan 3,3 y 5 voltios para alimentar dispositivos externos como sensores y actuadores. También incluye dos pines de tierra, ground (GND).

Puertos de E/S

La placa Arduino UNO posee 14 pines digitales que pueden ser configurados como pines de entrada o de salida. Seis de esos pines pueden configurarse como salidas PWM.

Existen también 6 pines analógicos que están conectados al convertidor ADC donde se pueden leer diferentes valores de tensión.

PIN	TIPO	ENTRADA/SALIDA	FUNCIÓN
0	Digital	Entrada/Salida	General / Recibir datos por el puerto serie (RX)
1	Digital	Entrada/Salida	General / Enviar datos por el puerto serie (TX)
2	Digital	Entrada/Salida	General / Interrupción externa
3	Digital	Entrada/Salida	General / Interrupción externa/ Salida PWM
4	Digital	Entrada/Salida	General
5	Digital	Entrada/Salida	General / Salida PWM

6	Digital	Entrada/Salida	General / Salida PWM
7	Digital	Entrada/Salida	General
8	Digital	Entrada/Salida	General
9	Digital	Entrada/Salida	General / Salida PWM
10	Digital	Entrada/Salida	General / Salida PWM / SS
11	Digital	Entrada/Salida	General / Salida PWM / MOSI
12	Digital	Entrada/Salida	General / MISO
13	Digital	Entrada/Salida	General / LED /SCK
A0	Analógico	Entrada	ADC
A1	Analógico	Entrada	ADC
A2	Analógico	Entrada	ADC
A3	Analógico	Entrada	ADC
A4	Analógico	Entrada	ADC
A5	Analógico	Entrada	ADC

Figura 4.13 Tabla de pines del Arduino UNO

4.2.2 Sensor ultrasónico

Objetivo y función

Una de las funcionalidades más importantes del robot es la de ser capaz de conocer la distancia a la que se encuentra de un objeto sin que exista ningún contacto físico entre ambos. Con esto, se pueden ubicar los distintos cuerpos que se encuentren en el entorno para:

- Ubicar los distintos objetos que se encuentra en el entorno
- Utilizar esta información para actuar de manera inteligente eligiendo una trayectoria u otra.
- Evitar obstáculos
- Representar la información obtenida en un mapa del entorno para poder visualizarla fácilmente

Para ello existen dos soluciones típicas y son los sensores por emisión de laser o los sensores por emisión de ultrasonidos.

En este caso se ha elegido un sensor de ultrasonidos ya que es capaz de ofrecernos unas medidas suficientemente precisas a un bajo coste.

Fundamentos

Un sensor de ultrasonidos es un sensor que emite ondas de ultrasonidos y las recibe. Está formado por uno o varios transductores que funcionan como emisor y/o receptor y un circuito de control y acondicionamiento de la señal.

Estas ondas de ultrasonido emitidas abarcaran una frecuencia fija a partir de los 20.000 Hz, que es la frecuencia máxima típica que puede percibir el oído humano. Las ondas de ultrasonido, como cualquier onda sonora, precisan de un medio físico para poder propagarse.

Los transductores emisores del sensor están formados por una membrana de un material piezoeléctrico que vibra al aplicarle una tensión y lo hace con una cierta frecuencia. El movimiento se transmite al medio que le rodea en forma de onda que se propagará por dicho medio como una onda.

Los transductores receptores también están formados por una membrana piezoeléctrica que al recibir el eco de la onda emitida generan una señal eléctrica que posteriormente será leída.

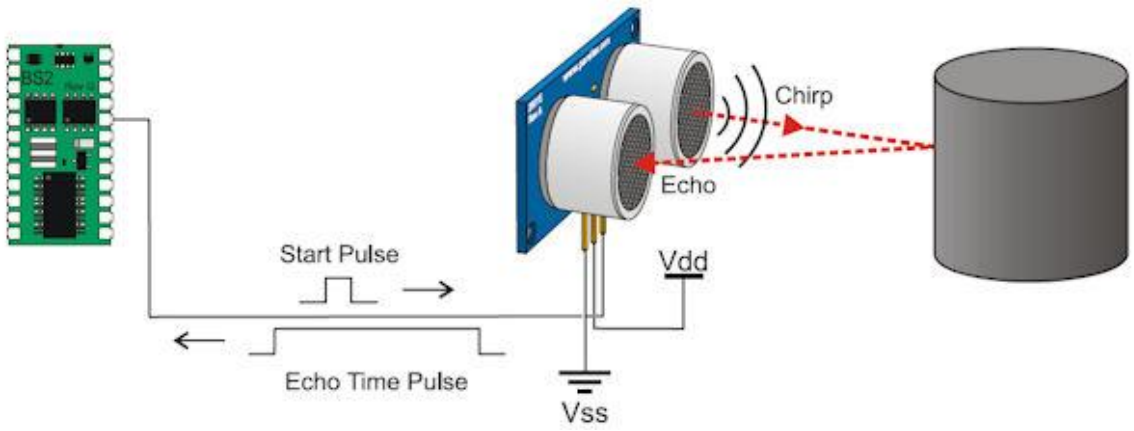


Figura 4.14 Funcionamiento de un sensor de ultrasonidos

La manera en que se determina la distancia del sensor a un objeto se realiza mediante la reflexión en dicho objeto y mediante el cálculo del tiempo de vuelo de la señal (Time of Fligth). Se medirá el tiempo que tarda la señal en viajar desde el transductor emisor hasta el objeto, rebotar y volver como un eco al receptor.

Conocido este tiempo y la velocidad de propagación de esta onda en el medio podemos calcular la distancia entre sensor y objeto según la siguiente fórmula. Donde D es la distancia al objeto, Vs la velocidad del sonido en el medio, y t el tiempo de vuelo.

$$D = \frac{Vs}{2} \times t \qquad [4.1]$$

Implementación

El sensor elegido es el HC-SR04. Este sensor es un modulo completo que está formado por una PCB con los circuitos de control y dos transductores. Uno es el emisor y otro el receptor. El modulo consta de 4 pines:

1. Alimentación(Vcc)
2. Tierra(GND)
3. Trigger
4. Echo

Figura 4.15 Sensor HC-SR04

El funcionamiento de este sensor es el siguiente.

1. Cuando se quiera hacer una medida, el pin de Trigger se debe poner a nivel alto durante al menos 10ms. El receptor se bloquea automáticamente para prevenir detecciones falsas por transmisión residual.
2. Posteriormente el circuito de control envía una señal de 8 pulsos con una frecuencia de 40kHz al transductor que emite la onda ultrasónica
3. La onda viaja a través del medio hasta que choque contra un cuerpo y rebote en sentido contrario al de la emisión dirigiéndose hacia el transductor de recepción.
4. Este eco hace vibrar la membrana del transductor de recepción y esto genera una señal eléctrica que posteriormente es amplificada por el circuito de control. Se escribe un nivel alto en el pin Echo que posteriormente se lee en el microcontrolador.
5. Mediante software, en el microcontrolador, se calcula el tiempo que ha transcurrido entre la activación de Trigger y la activación de Echo.
6. Una vez conocido el tiempo, sustituyendo los valores en la fórmula 4.1 se obtendrá el valor de la distancia.

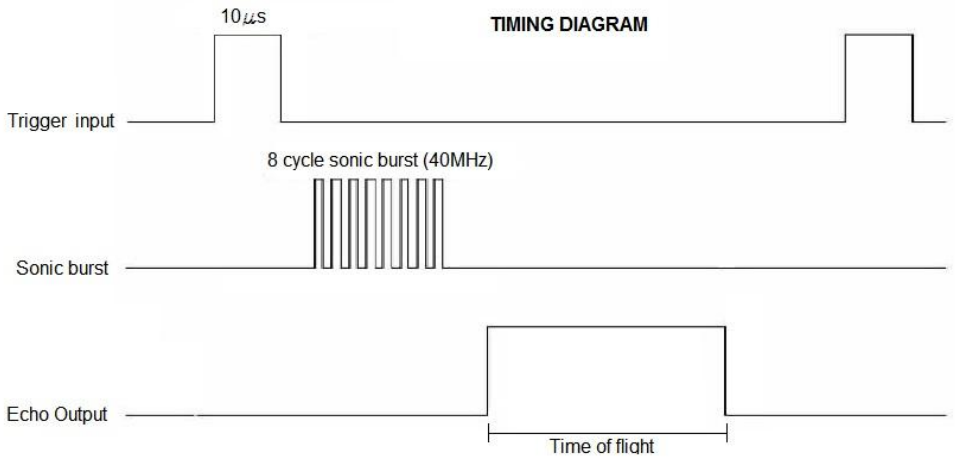


Figura 4.16 Diagrama lógico de funcionamiento del sensor HC-SR04
Fuente: Datasheet sensor HC-SR04

Los pines Vcc y GND del sensor se conectaran a 5 voltios y a tierra respectivamente.

Tanto el pin Trigger como el pin Echo se deben conectar a los pines digitales del microcontrolador para poder leer y escribir los niveles lógicos necesarios para el funcionamiento del sensor.

En este caso se ha conectado el pin Trigger al pin 7 y el pin Echo al pin 8.

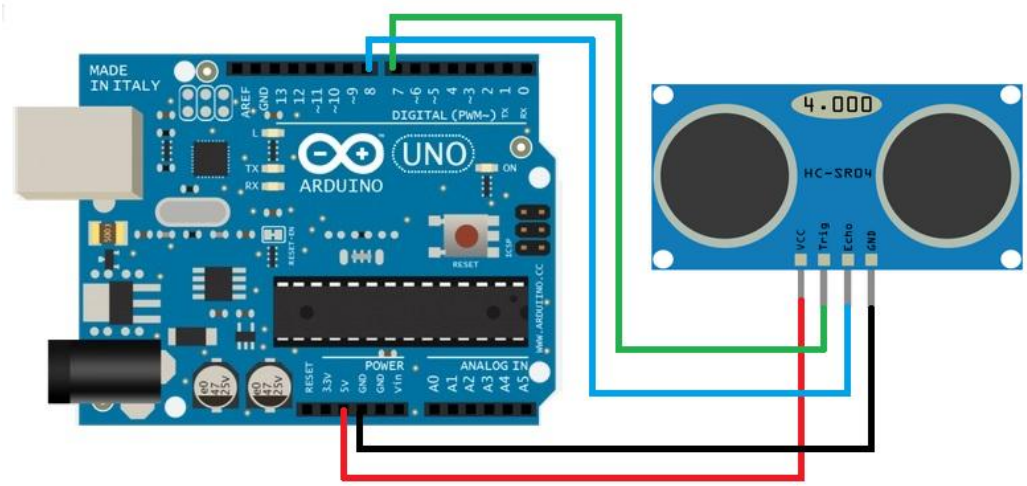


Figura 4.17 Esquema de conexiones del sensor ultrasónico HC-SR04 en un Arduino UNO

4.2.3 Sensor infrarrojo

Objetivo y función

Una funcionalidad deseada del robot es que pueda seguir objetos o paredes lateralmente cuando se esté moviendo. Los objetivos de la implementación de este sensor son:

- Detectar paredes u objetos laterales
- Estimar la distancia a la que se encuentran
- Esquivar o restringir movimiento si hay un obstáculo

Para cumplir estos objetivos se ha decidido utilizar un sensor infrarrojo montado en una placa de prototipado debido a su gran sencillez y bajo coste.

Fundamentos

Los sensores infrarrojos basan su funcionamiento en el efecto fotoeléctrico. Al chocar un fotón, con suficiente energía, contra una determinada superficie metálica, a la que se le aplica una tensión, este puede generar un par electrón-hueco y por lo tanto un flujo de electrones. Esta corriente se denomina fotocorriente.

Los sensores infrarrojos en el ámbito de la robótica se utilizan como sensores activos. Están formados por un emisor de radiación infrarroja y un receptor que pueden estar colocados en varias disposiciones.

Implementación

El circuito sensor está compuesto por un LED emisor de infrarrojos TSAL6200 del fabricante Vishay, un fotodiodo receptor de infrarrojos BPV10NF del mismo fabricante y dos resistencias, una de 220Ω y la otra de 10kΩ.

Existen 3 pines para su conexión:

- Tensión de entrada (5V)
- Tierra (GND)
- Tensión de salida (Vo)

El esquema del circuito diseñado se puede observar en la figura siguiente.

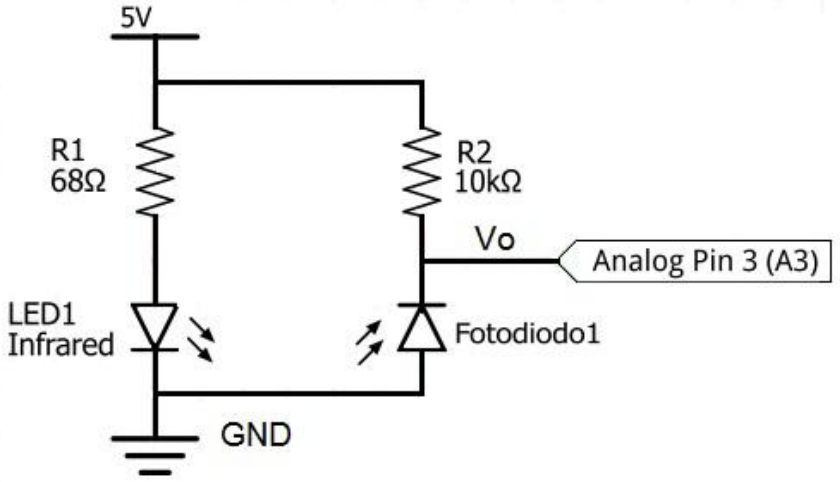


Figura 4.18 Esquema del circuito del sensor de infrarrojos

La salida del circuito la conectaremos directamente a un puerto analógico del microcontrolador donde posteriormente se leerán los valores a través del convertidor ADC.

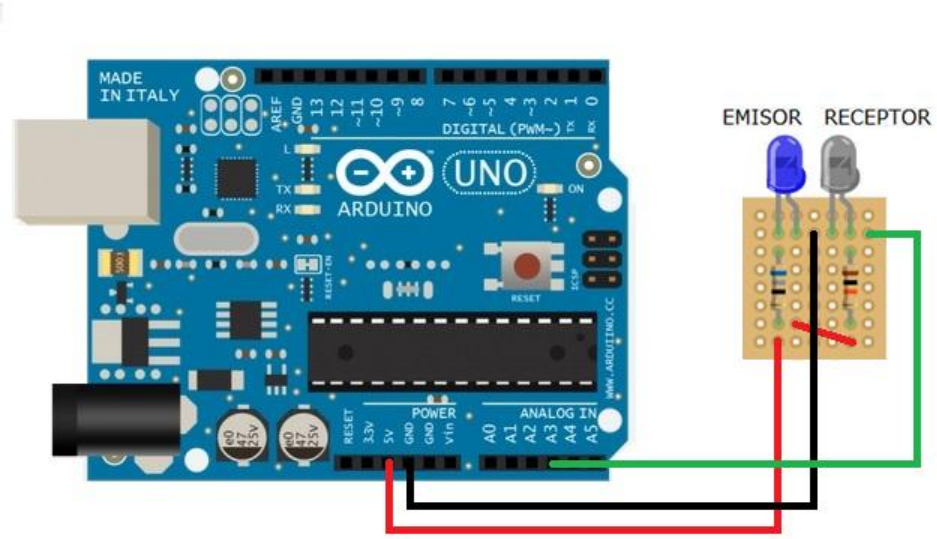


Figura 4.19: Esquema de conexiones del sensor de infrarrojo en un Arduino UNO

Dada la naturaleza del robot y la utilidad que se quiere dar al sensor, su funcionamiento debe ser por reflexión en objeto en el que el fotodiodo receptor recibe la radiación emitida por el emisor cuando esta rebota en un objeto próximo.

Un problema de diseño que se ha tenido con este sensor es que la luz ambiental del entorno en el que se encuentra interfiere con el fotorreceptor y los valores registrados en la salida varían dependiendo de la luminosidad del entorno.

Para reducir la influencia existen se ha decidido cubrir el receptor con un material opaco dejando solo una abertura por donde pueda entrar la radiación del LED emisor evitando la que le pueda llegar desde arriba o por los laterales desde otras fuentes.

Cálculos

A continuación se muestran los cálculos realizados para diseñar el sensor en función de las características del emisor y el receptor que ofrecen sus datasheet.

Para calcular la resistencia R que hay que incluir en la rama del LED emisor se ha utilizado la siguiente formula. Donde V_{cc} es la tensión de alimentación, 5 voltios; V_{LED} es la tensión directa en el LED emisor, 1,6 voltios e I_{MAX} que es la corriente máxima recomendada por el LED, 100 miliamperios.

$$R = \frac{V_{cc} - V_{LED}}{I_{MAX}} \quad [4.2]$$

Sustituyendo los valores, la resistencia mínima que asegura el funcionamiento del LED emisor es de 34 ohmios.

El problema es que con esta resistencia la corriente es de 100 miliamperios, por lo que el consumo es demasiado elevado. Para disminuirlo sin afectar la funcionalidad del sensor se ha escogido una resistencia de 68 ohmios la cual permite una corriente a través del LED de 50 miliamperios y es suficiente para que este emita con suficiente intensidad.

Caracterización del sensor

Debido a que se ha construido el sensor y no se ha adquirido a un fabricante se ha decidido incluir una grafica que muestra el comportamiento del sensor.

En la siguiente grafica (Figura 4.20) se ha representado el valor leído en el microcontrolador a través del ADC respecto a la distancia. Las medidas se han tomado varias veces y con materiales y luminosidades diferentes.

Los materiales son azulejo cerámico y madera. Se han escogido esos porque son materiales típicos en interiores y el azulejo refleja mucho y la madera no. Las medidas han sido tomadas en interior con luz artificial y sin luz.

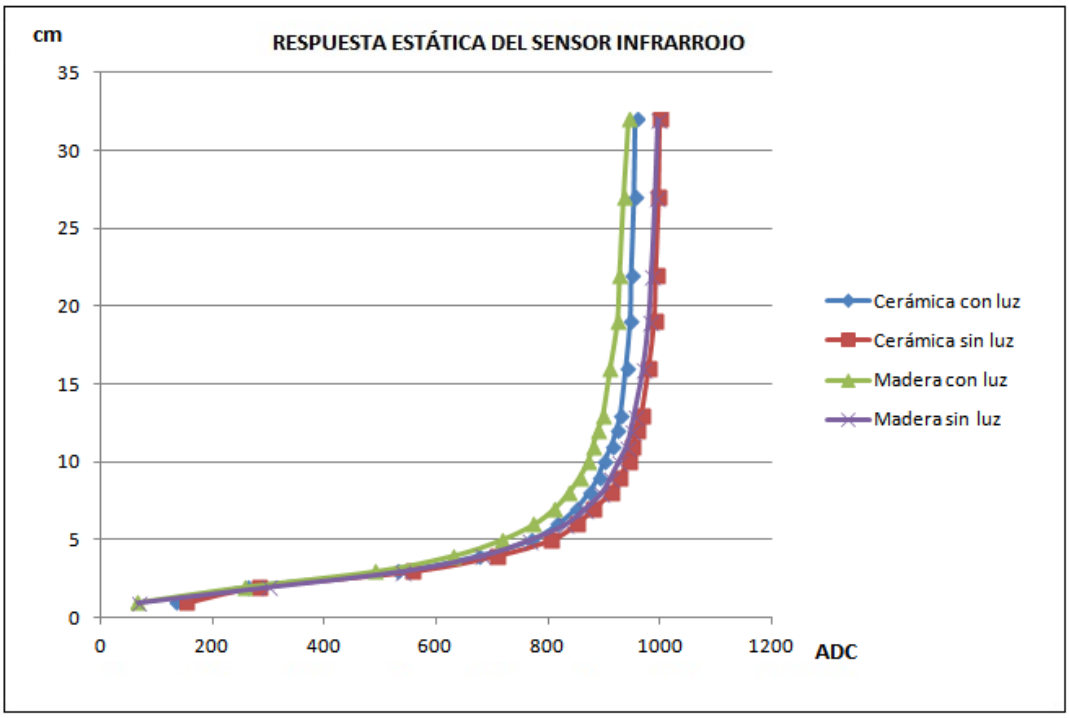


Figura 4.20 Respuesta estática del sensor infrarrojo

En la grafica se puede observar que los valores registrados por el sensor en ausencia de luz son casi idénticos para los dos materiales.

En la presencia de luz artificial también se asemejan entre ellas pero difieren un poco de las medidas sin luz. Aun así, la respuesta del sensor sigue una curva muy bien definida y, bastante similar en cada caso.

4.2.4 Acelerómetro

Objetivos y función

La intención de incluir un acelerómetro en el robot no es medir sus aceleraciones sino medir su inclinación. Esta inclinación se medirá en ángulos y se podrá expresar con dos componentes:

- El ángulo de inclinación frontal.
- El ángulo de inclinación lateral.

Se necesitará un sensor de 3 ejes (X, Y, Z) para poder calcular estos dos ángulos.

Fundamentos

Un acelerómetro es un dispositivo que permite medir las aceleraciones lineales a las que están sometidos sus ejes. Se suele utilizar embebido dentro de un sistema para conocer las aceleraciones que sufre el cuerpo en las 3 direcciones del espacio, por ejemplo para medir las vibraciones, choques y fuerzas inerciales en un vehículo.

Dentro de las diferentes tecnologías de fabricación de acelerómetros, los capacitivos son muy comunes debido a su sencillez de uso y precio ya que suelen estar integrados en un MEMS (Micro Electromechanical Systems) incorporando ya un circuito de acondicionamiento de la señal.

El principio de funcionamiento de este tipo de acelerómetros está basado en la Ley de Hook y en la segunda Ley de Newton a través de las cuales se puede expresar la aceleración en función del desplazamiento.

$$a = \frac{k}{M} \cdot \Delta x \qquad [4.3]$$

Dentro del sensor se tiene una masa inercial sensible a la aceleración y una serie de condensadores diferenciales. El desplazamiento de esta masa provoca un cambio en la capacidad de los condensadores que se puede registrar en una señal de salida.

Aplicando ciertas formulas trigonométricas a los valores de estas salidas se pueden hallar los ángulos de inclinación respecto el plano terrestre a los que está sometido el sensor, para lo cual se va a implementar.

Implementación

El sensor que se ha utilizado en esta aplicación es el ADXL335 del fabricante Analog Devices.

Figura 4.21 Acelerómetro ADXL335
Fuente: http://www.jayconsystems.com/adxl-335-triple-axis-accelerometer.html

Este sensor contiene un acelerómetro capacitivo de 3 ejes con un rango de medida de -3g a 3g y de bajo consumo. En sus salidas, genera una señal analógica ya acondicionada (Figura 4.21) y proporcional a la aceleración correspondiente a cada uno de sus ejes.

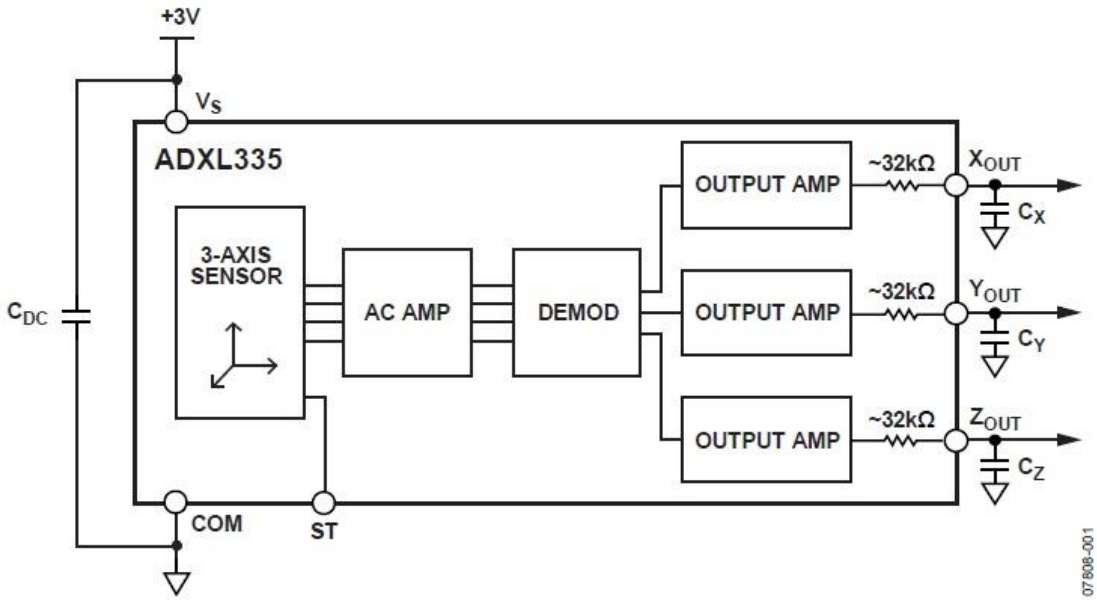


Figura 4.21 Diagrama del sensor ADXL335
Fuente: Datasheet ADXL335

El sensor tiene 5 pines:

1. Alimentación (3V)
2. Tierra (GND)
3. Eje X
4. Eje Y
5. Eje Z.

La alimentación del sensor es a 3 voltios pero puede llegar a un máximo de 3,6 voltios, por lo que el pin de alimentación se podrá conectar a los 3,3 voltios del microcontrolador. El pin de tierra se conectará al pin GND del microcontrolador.

Los pines X, Y, Z se conectaran a las entradas analógicas A0, A1, A2 del microcontrolador donde se leerán los valores que proporciona el sensor.

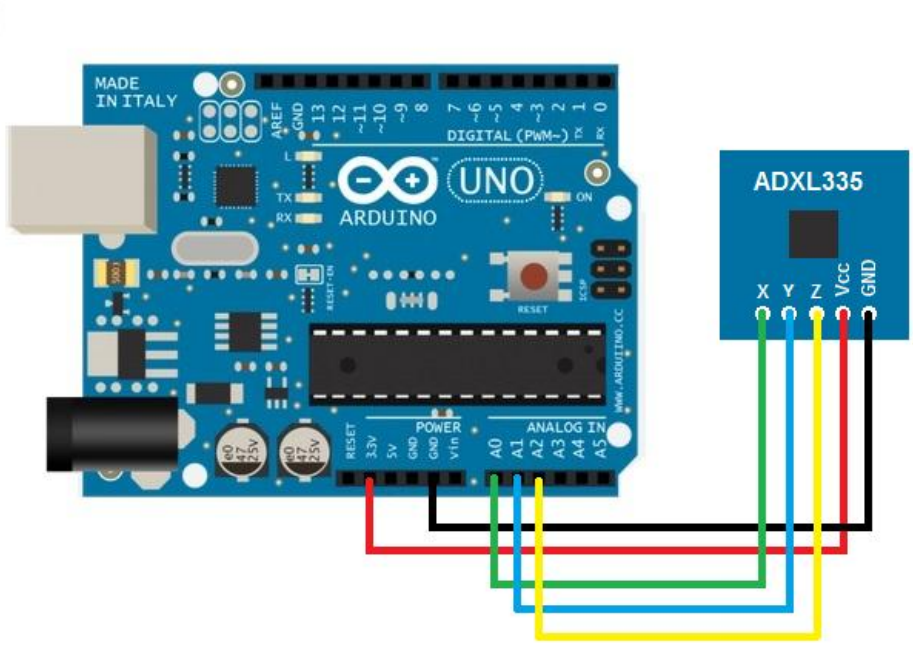


Figura 4.22: Esquema de conexiones de un ADXL335 en un Arduino UNO

La ubicación del sensor deberá ser en una superficie fija paralela a la base del robot para poder obtener medidas fiables de la inclinación.

4.2.5 Encoders

Objetivos y función

Los encoders de un robot son un elemento esencial se quiere realizar un buen control de los motores del robot y un cálculo de su odometría.

Los objetivos a conseguir son:

- Contar los pasos que realiza el disco codificado
- Calcular la velocidad de giro de las ruedas

Fundamentos

Un encoder es un dispositivo electromecánico que permite conocer la posición y velocidad de un objeto que se desplace linealmente o rote sobre un eje.

En este trabajo se han utilizado dos encoders con las siguientes características:

- Incremental
- De tipo óptico por infrarrojos
- Detección por interrupción de barrera.

Un encoder óptico utiliza la interrupción de haces de luz para detectar el desplazamiento de un disco que gira solidario a un eje. Posee un fotointerruptor compuesto por un LED emisor en un extremo y un fotorreceptor en el otro. Al ser de tipo incremental, el disco posee unas ranuras a través de las cuales puede pasar el haz de luz que emite un LED y ser detectado por el fotoreceptor en el otro extremo.

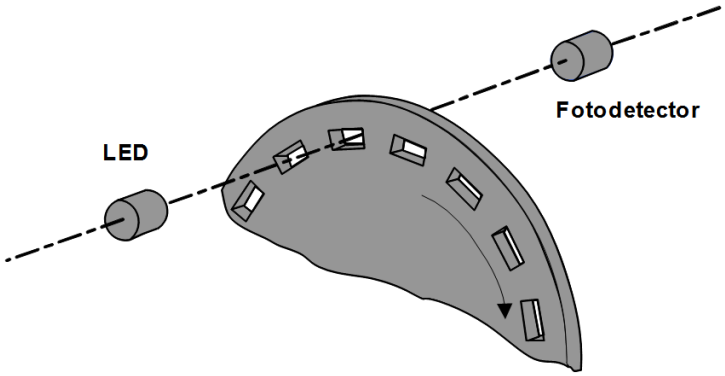


Figura 4.23 Encoder óptico en barrera
Fuente: Pérez García, M. A., Álvarez, J. C., & Campo, J. C. Instrumentación electrónica.

Esta disposición que se observa en la figura anterior se denomina "en barrera" ya que, al girar el disco entre el emisor y el receptor, corta o deja pasar el haz de luz habilitando o interrumpiendo la detección en el fotorreceptor.

La lectura que proporciona el encoder son los pulsos obtenidos en el fotorreceptor (Figura 4.24). Si se cuentan los pulsos en un intervalo de tiempo se puede obtener la posición del disco y la velocidad de giro.

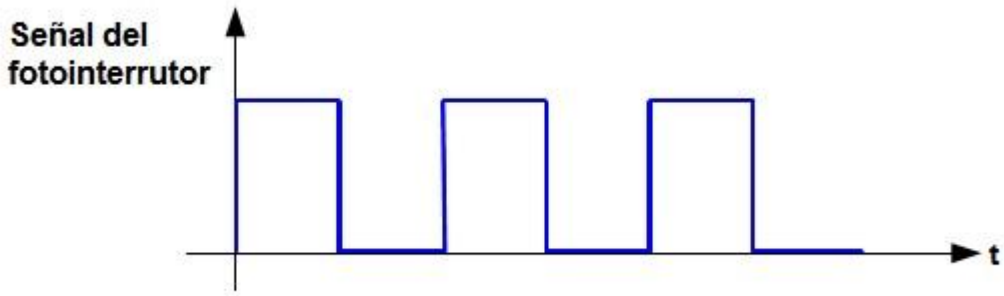


Figura 4.24 Señal recibida del fotointerruptor

Implementación

Para el robot se han incluido dos encoders, uno en cada rueda motriz.

Esto se suministran comercialmente como un modulo completo [tipo] y cada uno posee 3 pines:

- Alimentación (5V)
- Tierra (GND)
- Salida (Vo)

El pin Vo es el que proporciona la lectura del encoder dando un nivel alto cuando el haz atraviesa y un nivel bajo cuando el haz no pasa.

El pin de alimentación del los encoders se conectará al pin de 5 voltios del microcontrolador, el de tierra al GND y el de Vo al pin 2 y al 3 respectivamente.

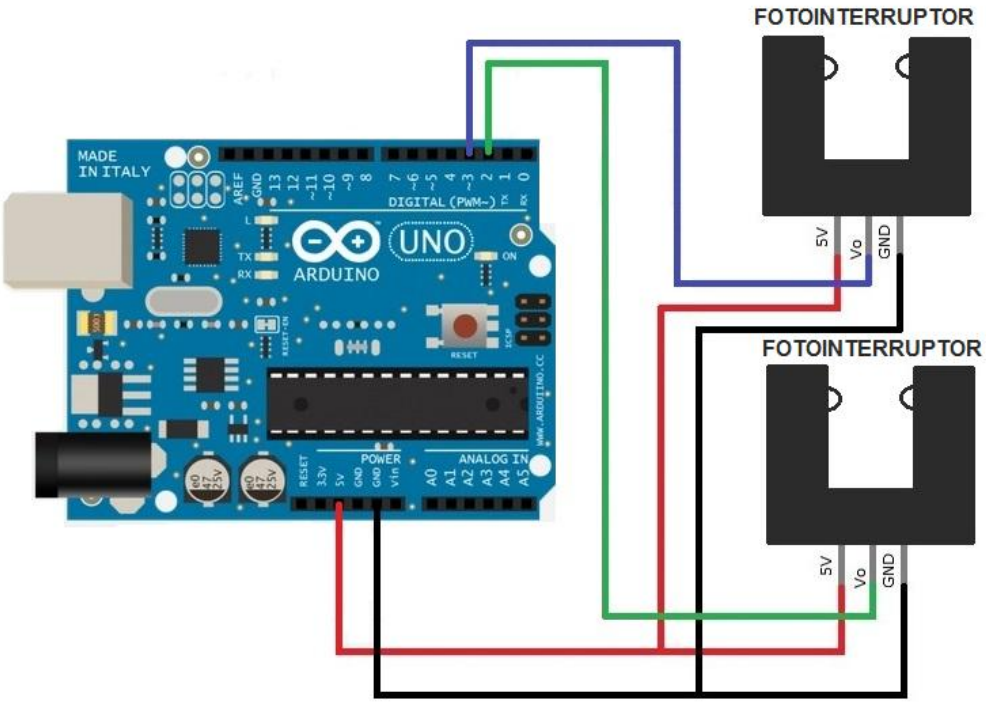


Figura 4.25 Esquema de conexiones de dos fotointerruptores en un Arduino UNO

Se han elegido los pines 2 y 3 porque estos pines permiten gestionar interrupciones externas que permitirán el conteo de pulsos. Esta decisión se debe a que el conteo de pulsos del encoder debe hacerse mientras se ejecuta el programa principal del robot para así obtener unas medidas precisas de la posición en cualquier instante de tiempo.

El disco con las ranuras es un disco de plástico ABS que se fija mecánicamente al motor el cual girará entre el emisor y el receptor. La posición de este disco tiene que ser muy precisa para que el haz de luz pase exactamente por las ranuras.

4.2.6 Servomotores

Objetivos y función

Los motores que mueven las ruedas del robot son dos servomotores de rotación continua.

Para el movimiento del robot podrían haberse usado:

- Dos motores DC que generan el movimiento.
- Cajas reductoras para elevar el par de giro.
- Circuito de control para poder invertir la dirección de giro.

Se ha decido usar servomotores porque ofrecen una solución completa al incluir todos los componentes en un mismo modulo.

Fundamentos

Un servomotor de rotación continua es un sistema electromecánico formado por un motor de corriente continua, una caja reductora, y un circuito de control.

Figura 4.26 Componentes de un servomotor
Fuente: http://cocherc.wordpress.com

La finalidad de un servomotor de rotación continua es transmitir un movimiento rotatorio constante pero regulable a un eje motriz.

Suele tener tres entradas:

1. El cable de alimentación
2. El cable de tierra
3. El cable de control.

El servomotor se controla con una señal PWM. Esta señal entra al circuito de control y suministra un voltaje efectivo al motor para que este gire.

Una señal PWM es una señal periódica formada por una cadena de pulsos con una anchura determinada. El control de un servomotor reside en modificar esta anchura.

Según el ancho de pulso que recibe el servomotor la velocidad de giro será mayor o menor.

Típicamente, la frecuencia de la señal PWM es de 50Hz por lo que el periodo es de 20 milisegundos. Cuando el ancho de pulso de la señal PWM es de 1,5 milisegundos el servo no se mueve, se encuentra en su posición de reposo. Si el ancho de pulso aumenta hasta 2 milisegundos, el servomotor gira a su

máxima velocidad en un sentido. Si el ancho es de 1 milisegundo, el servomotor girara a máxima velocidad en sentido opuesto.

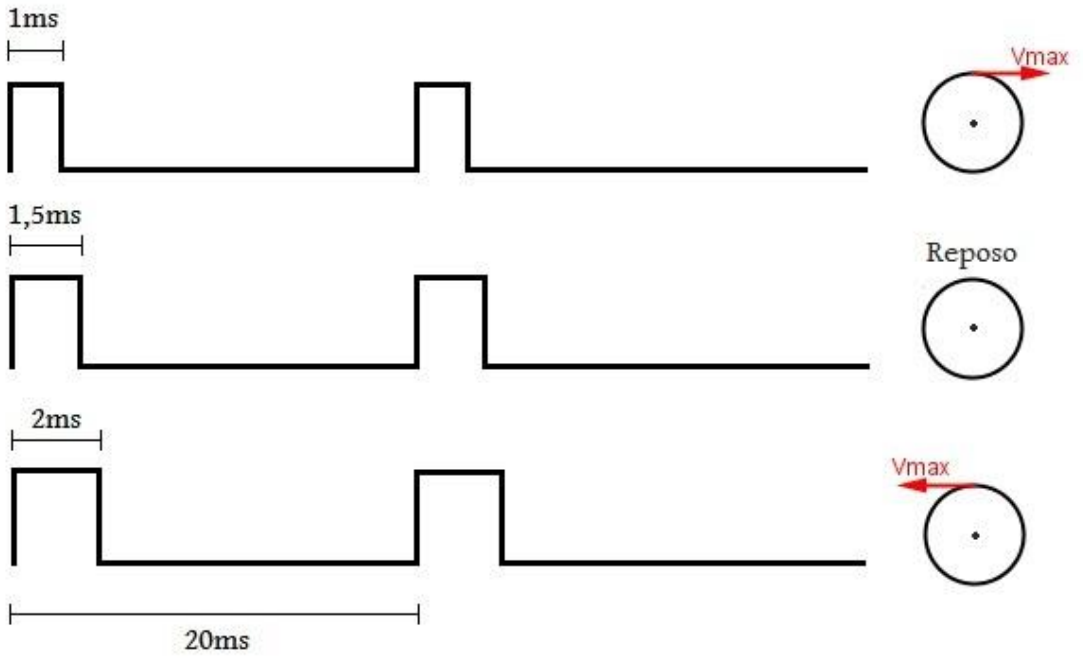


Figura 4.27 Señales PWM de control de un servomotor de rotación continua

Cualquier velocidad entre los valores máximos puede conseguirse modificando este ancho de pulso. Sin embargo la relación entre el ancho de pulso y la velocidad de giro no suele ser lineal.

Implementación

El modelo de servomotor elegido es el S4309R de la marca SpringRC.

Figura 4.28 Servomotor S4309R
Fuente: SpringRC - http://www.springrc.cn

Se ha elegido este modelo porque tiene un precio asequible y proporciona un par elevado y una velocidad máxima suficiente para el movimiento del robot.

Los servomotores se alimentaran a una tensión máxima de 6 voltios, esta es la tensión a la que el servomotor es capaz de proporcionar mayor par y mayor velocidad como se observa en la siguiente figura.

Products specification								Technical parameters						
Size (mm)					Weight		Wire	4.8V			6V			Rotation angle
								Speed	Torque		Speed	Torque		
A	B	C	D	E	g	oz	cm	rpm	kg·cm	oz·in	rpm	kg·cm	oz·in	
41.3	20.7	40.2	50.3	10.0	56	1.98	30.0	43	7.7	107.12	57	8.7	121.03	360°

Figura 4.29 Características del servomotor SpringRC S4309R
Fuente: SpringRC - http://www.springrc.cn

La alimentación del servomotor es un aspecto importante del diseño del robot. Se sabe que el servomotor necesita alimentarse a 6 voltios para ofrecer el máximo par de giro, pero el microcontrolador solo proporciona 5 voltios como máximo.

El servomotor puede alimentarse a esta tensión pero surge otro problema y es que la corriente que circula por él, a pleno rendimiento, esta entorno a 1 y 1,5 amperios, lo que puede dañar el microcontrolador si se le hace pasar las dos corrientes de cada servo por sus pines.

Por ello, se ha decidido implementar un circuito externo con una batería y un regulador de voltaje que proporcione 6 voltios en su salida para alimentar a los servomotores. Este circuito se explica con más detalle en el siguiente apartado.

Los servomotores tienen 6 pines que se deben conectar, 3 cada uno. Cada servo tiene:

- Pin de alimentación que se conecta a los 6V que proporciona el regulador de tensión
- Pin de tierra que se conecta a la tierra del regulador
- Pin de control que se conecta a un pin digital del microcontrolador.

El pin de control debe conectarse a los pines 5 y 3 del microcontrolador ya que son capaces de generar la señal PWM necesaria.

4.2.7 Regulador de tensión y batería

Objetivos y función

Implementar una batería en el robot es esencial dada su naturaleza móvil ya que debe ser alimentado sin necesidad de estar conectado a una red de suministro eléctrico externa. Esta batería necesita un sistema de adaptación de tensión para alimentar correctamente a los diferentes componentes. Se ha decidido implementar un regulador te tensión lineal.

La batería que se ha decidido usar es una batería de polímero de litio ya que ofrece una tensión mucho más constante durante su uso que una de NiCd (Níquel-Cadmio) y su recarga es suele ser más rápida.

Fundamentos

Las baterías LiPo (Lithium Polimer) que se comercializan suelen estar formadas por un conjunto de celdas las cuales ofrecen un voltaje nominal de 3,1V. Por lo tanto las baterías que se pueden adquirir típicamente poseen de tensiones múltiplos de esta cantidad, por ejemplo 7,4V y 11,1V.

Existen diferentes capacidades dentro de cada tensión disponible, esta capacidad se suele expresar en miliamperios por hora (mAh). Esta unidad expresa la corriente máxima que puede suministrar la batería durante una hora.

Este tipo de baterías suelen incluir dos cables:

- Cable de alimentación. Es el cable que se conecta al circuito que se desee alimentar
- Cable de carga. Es el cable que se conecta al cargador de la batería y permite una carga balanceada de cada celda.

Un regulador de tensión es un circuito electrónico que convierte una tensión de entrada a una tensión de salida fija que puede ser regulable.

Los reguladores de tensión son muy utilizados en todos los ámbitos de la electrónica de potencia y suelen encontrarse en fuentes de alimentación de cualquier sistema electrónico. Estos reguladores pueden estar basados en diferentes tecnologías pero existen dos grupos diferenciados que son: reguladores lineales y conmutados.

La ventaja de los conmutados frente a los lineales es que son mucho más eficientes ya que tienen menores perdidas por disipación de calor, las desventajas son que su complejidad es mayor y son caros.

Normalmente la elección de un tipo de regulador u otro está definida por el tipo de aplicación en la que va a estar integrado y los requerimientos de dicha aplicación a nivel electrónico.

Implementación

Para este robot se ha adquirido una batería LiPo de 7,4 voltios y 1000mAh de la marca Turnigy modelo 1.0 esta batería está formada por dos celdas de 3,7 voltios conectadas en serie.

Figura 4.30 Bateria Turnigy 1.0
Fuente: http://www.turnigy.com

La batería puede conectarse directamente al microcontrolador mediante un conector de tipo Jack ya que este admite entre 5 y 16 voltios de alimentación.

El problema surge cuando se ha de alimentar a los servomotores. Debido a que los servomotores deben hacerlo como máximo a 6V necesitamos reducir el voltaje de salida de la batería a 6V para no dañarlos. Para ello se va a diseñar el circuito regulador de tensión.

Para este robot se ha decidido utilizar un regulador de tensión lineal. En concreto el LM7806.

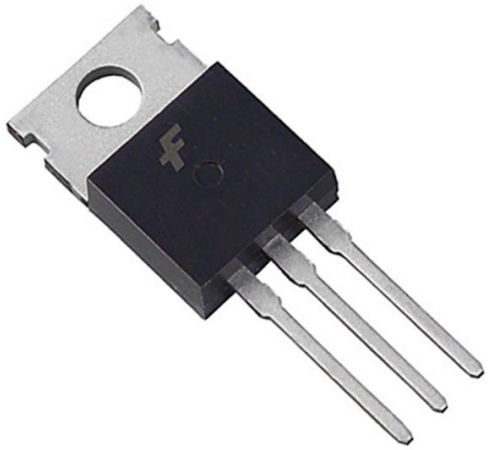


Figura 4.31: Regulador LM7806

Como se puede observar, este regulador tiene 3 pines:

1. Tensión de entrada (Vin)
2. Tierra (GND)
3. Tensión de salida (Vout)

Este regulador se implementara en una configuración de Fixed Output que proporciona una tensión de salida típica de 6 voltios.

Para esta configuración hay que conectar el pin de entrada a la fuente de alimentación, en nuestro caso a la batería. El pin de tierra al negativo de la batería y el pin de salida a la alimentación de cada servomotor.

Para obtener una salida estable se conectan dos condensadores en paralelo, uno entre la entrada y tierra y otro entre la salida y tierra. Ambos de 0,33μF.

Para completar las conexiones hay que conectar todas las tierras entre ellas. Es decir, los pines negativos de cada servomotor a la tierra del regulador.

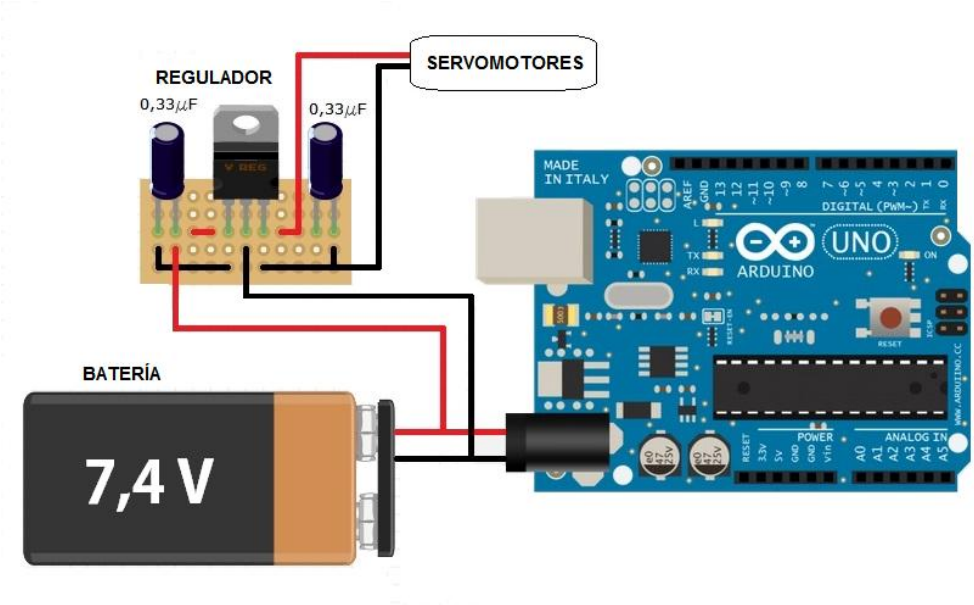


Figura 4.32 Esquema de conexiones de la batería y regulador al Arduino UNO

4.2.8 Modulo Bluetooth

Objetivos y función

Uno de los objetivos principales a conseguir en el robot es la comunicación con otro dispositivo, en este caso un teléfono móvil, al que le pueda enviar información relevante sobre su estado actual y a través del cual pueda recibir comandos para realizar las diversas acciones.

Como requisitos fundamentales se tiene:

- La comunicación debe ser inalámbrica
- Debe existir comunicación aunque no exista una línea visual entre emisor y receptor.

El dispositivo que permite llevar a cabo esta comunicación desde el robot es un modulo Bluetooth que se conectara al microcontrolador. En el teléfono móvil se utiliza el propio Bluetooth que viene incorporado de serie.

Fundamentos

El modulo Bluetooth que se va a utilizar funciona como una interfaz del puerto serie del microcontrolador.

La información que se envíe por el puerto serie (Tx) será procesada y emitida por el modulo a través del protocolo Bluetooth. A su vez, toda la información que le llegue al modulo vía Bluetooth se procesara y se escribirá en el puerto serie (Rx) para poder leerla después.

Este tipo de módulos permite la conexión entre diferentes microcontroladores que tengan un protocolo de comunicaciones UART, permitiendo asi la conexión entre Arduino y un teléfono móvil, tablet, PC, etc.

Implementación

El modulo implementado es el HC-05 ya que puede funcionar como maestro o como esclavo y su coste es muy reducido.

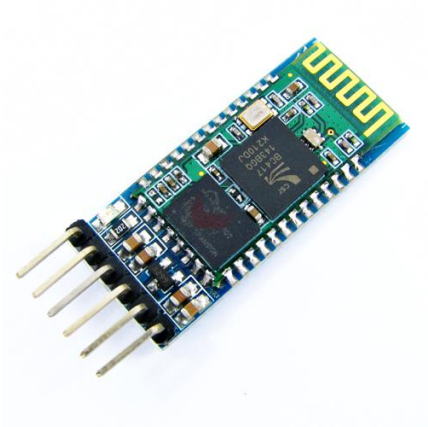


Figura 4.33 Modulo Bluetooth HC-05

Este modulo tiene 6 pines de conexión:

- Alimentación(5V)
- Tierra(GND)
- Tx
- Rx
- State
- Key

El pin de alimentación y tierra se conectaran al microcontrolador en los pines de 5V y GND respectivamente. El pin Rx será donde el modulo recibe los datos de salida desde el puerto serie, por lo tanto se conectara al pin 0 (Tx) del microcontrolador.

El pin Tx del modulo será donde envía los datos recibidos vía Bluetooth escribiéndolos en la entrada del puerto serie, por lo tanto se conectara al pin 1 (Rx) del microcontrolador.

4.3 Localización del robot y odometría

En este apartado se definen las herramientas matemáticas que permiten localizar al robot y están basadas en la cinemática de un robot diferencial que se ha estudiado en el apartado 3.2.

El robot parte de una posición inicial en la que su centro geométrico se encuentra en el punto $(x0, y0)$ y el sistema de referencia fijo XY está alineado con el móvil $X'Y'$ formando el eje Y' con el eje X un ángulo de 90º.

El robot al moverse recorre una distancia D y el ángulo que forma la recta D con el eje X es α.

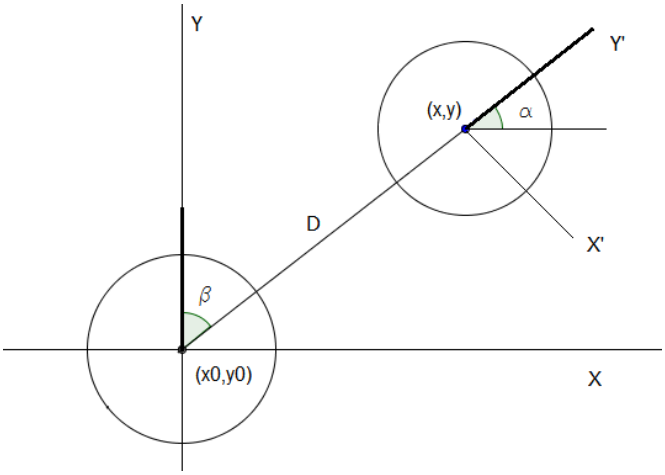


Figura 4.34 Posición del robot

Por lo tanto la posición del robot está definida por las siguientes formulas.

$$x = x0 + D \cdot \cos\ (\alpha) \quad [4.4]$$

$$y = y0 + D \cdot \sin\ (\alpha) \quad [4.5]$$

Los cálculos de los parámetros D y α se realizan a través de la distancia recorrida por las ruedas según el tipo de movimiento en los siguientes apartados.

4.3.1 Movimiento rectilíneo

Cuando el robot avanza lo hace de manera rectilínea, esto conlleva que sus ruedas giren a la misma velocidad. Por lo tanto la distancia que recorren es la misma.

Suponiendo que no existe deslizamiento entre la rueda y el suelo, esta distancia D se puede definir por el número de pulsos que detecta el fotointerruptor del encoder n, por la distancia que hay entre marcas, Dm.

$$D = Dm \cdot n \quad [4.6]$$

Conociendo que la rueda tiene un diámetro de 6,4 centímetros, se sabe que su perímetro es de 20,106 centímetros y sabiendo que el encoder tiene 8 marcas, la distancia entre ellas será 2,513 centímetros.

$$Dm = \frac{20{,}106\ cm}{8\ marcas} = 2{,}513\ cm \quad [4.7]$$

Por lo tanto la distancia recorrida D se puede expresar en función del número de pulsos detectados n, como:

$$D = 2{,}513 \cdot n \qquad [4.8]$$

4.3.2 Rotación hacia la izquierda

Cuando el robot realiza una rotación lo hace sobre su eje central. La distancia que recorre su centro de masas es nula pero no la de sus ruedas. En consecuencia el robot gira un determinado ángulo.

Para calcular el ángulo de giro del robot se va a suponer que no existe deslizamiento entre las ruedas y la superficie.

El mínimo ángulo de giro que se puede medir se realiza cuando una de las ruedas del robot avanza una marca del encoder en un sentido y la otra en el sentido contrario.

La distancia recorrida por cada rueda según el apartado anterior es de 2,513 centímetros lo que equivale a un giro de 15 grados.

Por lo tanto el ángulo que gira el robot, β podra expresarse en función de los pulsos recibidos por el fotointerruptor n, como:

$$\beta = 15 \cdot n \qquad [4.9]$$

4.3.3 Rotación hacia la derecha

La fórmula para calcular los grados girados hacia la derecha es la misma que para una rotación hacia la izquierda solo que con sentido negativo.

$$\beta = -15 \cdot n \qquad [4.10]$$

4.3.4 Calculo de alfa

El ángulo β que gira el robot es una variable útil para controlarlo pero a la hora de posicionarlo en el espacio la variable que se necesita hallar es α.

El cálculo de α es sencillo, conocidos el ángulo de giro del robot β y el ángulo inicial de α (figura anterior), la expresión es la siguiente:

$$\alpha = 90 + \beta \qquad [4.11]$$

4.4 Programación del microcontrolador

En esta sección se describirá el código que se ha implementado en el microcontrolador para controlar el robot. Primero se realizara una descripción de la manera de programar un Arduino y luego se entrará a desarrollar el diagrama de clases que se ha diseñado para controlar el robot.

Para cargar un programa en el microcontrolador del Arduino se utiliza el puerto USB que dispone la placa y se conecta a un PC donde se haya instalado el entorno de desarrollo de Arduino o en ingles, IDE (Interactive Development Enviroment).

Este IDE permite escribir el programa utilizando el lenguaje C y C++. Una vez finalizado lo compila, ensambla y carga en el microcontrolador.

La programación que se ha realizado es una programación orientada a objetos (POO). Este tipo de programación permite hacer un programa modular fácilmente modificable, depurable y ampliable. Además, al relacionar las diferentes clases que lo componen con elementos hardware del propio robot, facilita añadirle nuevos sensores y actuadores haciéndolo muy flexible.

La siguiente figura muestra un diagrama de clases completo del software del microcontrolador. Posteriormente se va describiendo cada clase y sus métodos más importantes.

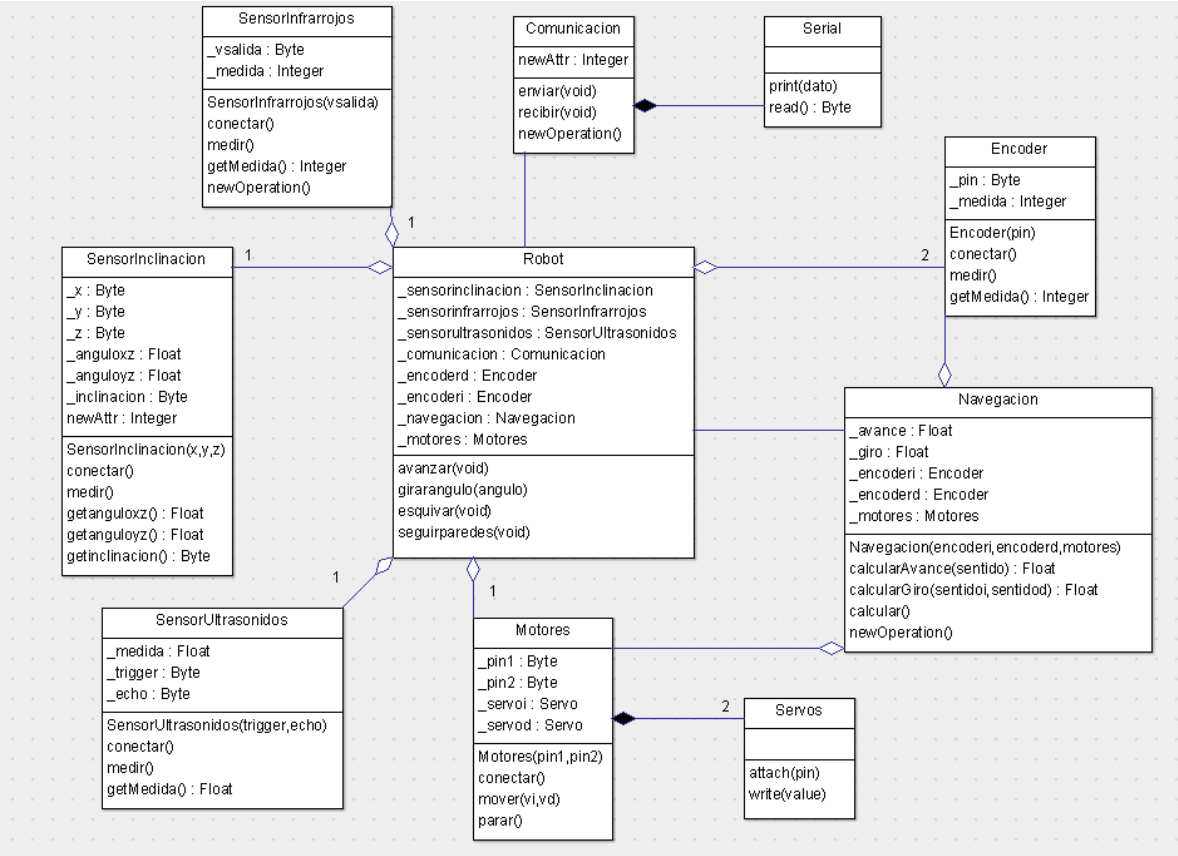


Figura 4.35 Diagrama de clases

4.4.1 Funciones y métodos importantes

Las siguientes funciones están definidas en la librería `Arduino.h` y son las que más se han utilizado en la programación del microcontrolador.

- `pinMode (pin, OUTPUT/INPUT)` Permite configurar un puerto digital como entrada o salida.
- `digitalWrite (pin, HIGH/LOW)` Permite escribir un 1 o un 0 lógico en un pin de salida.
- `delay (int x)` Permite parar la ejecución del programa durante `x` milisegundos.
- `pulseIn(pin,HIGH/LOW)` Devuelve el tiempo en milisegundos que ha transcurrido desde un nivel alto o bajo al siguiente.
- `analogRead(byte x)` Devuelve la lectura del ADC del pin `x`.

- `float atan (float x)` Devuelve una variable tipo float que es el valor de la arcotangente de x.
- `write (float x)` Método de la clase Servo. Modifica la señal PWM de control de un servomotor. Los valores límite funcionales de x son 0 y 180.
- `attachInterrupt(x, Func, FALLING/RISING/CHANGE)` Configura una interrupción externa en el pin x. Cada vez que se lea un flanco de subida `RISING` , bajada `FALLING` o ambos `CHANGE` se ejecuta la función `Func`.

4.4.2 SensorUltrasonidos

Se ha declarado `SensorUltrasonidos` como una clase agregada a `Robot`. Los atributos de esta clase son: los números de los pines digitales a los que se han conectado la entrada y la salida del sensor, `_trigger` y `_echo`, y una variable tipo `float` denominada `_medida` donde se guarda el ultimo valor leído por el sensor.

Los métodos importantes de esta clase son los siguientes.

void conectar (void)

Con este método se configuran los pines digitales del microcontrolador para poder leer o escribir por ellos.

En el caso de este sensor, el pin que se conecta a Trigger debe ser de salida por lo que se declara como `pinMode(_Trigger, OUTPUT)`.

El pin que se conecta a Echo debe ser de entrada por lo que se declara como `pinMode(_echo, INPUT)`.

void medir (void)

Con este método se realiza la medida de la distancia y se guarda en el atributo `_medida`.

Primero se escribe un nivel alto en el pin de Trigger para emitir la onda ultrasónica. Este pulso tiene que tener una duración de 10 milisegundos.

Se utilizara la función `digitalWrite(_trigger, HIGH)` y se utilizara `delay (10)` para realizar esta espera.

Después de los 10 milisegundos se desactiva utilizando la misma función pero esta vez `digitalWrite(_trigger, LOW)`.

Cuando en el pin Echo del microcontrolador llegue un nivel alto significa que la onda ha rebotado, la ha detectado el transductor receptor y el circuito de acondicionamiento se ha puesto a nivel alto.

Se utiliza la función `pulseIn (pin, HIGH/LOW)` para calcular el tiempo que ha tardado la onda en viajar.

Posteriormente se utiliza la Formula 4.1 para obtener la distancia deseada.

Una vez finalizado este algoritmo se guarda el valor obtenido en el atributo `_medida` de la clase.

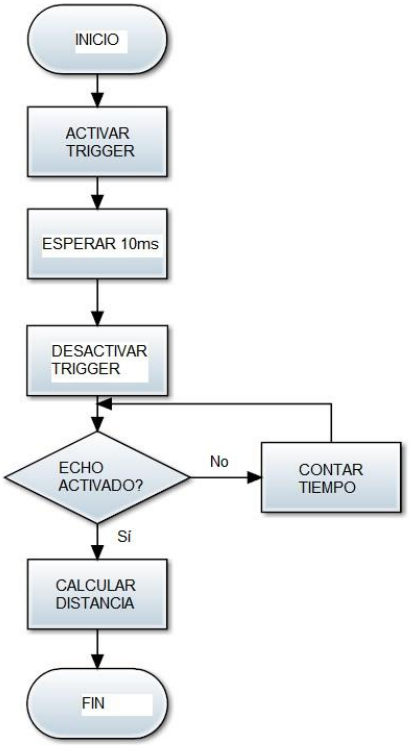


Figura 4.36 Diagrama de flujo del sensor ultrasónico

4.4.3 SensorInfrarrojos

En el software se ha declarado el `SensorInfrarrojos` como una clase agregada a `Robot`. El atributo de esta clase es el número del pin analógico al que se ha conectado la salida del sensor `_salida` y una variable `float _medida` donde se guarda el último valor leído por el sensor.

Los métodos importantes de esta clase son los siguientes

void conectar (void)

Con este método se configura el pin analógico del microcontrolador para poder leer valores a través del convertidor analógico-digital.

En este caso, el pin al que se conecta la salida del sensor se requiere que sea de entrada por lo que se declara como `pinMode(_salida, INPUT)`.

void medir (void)

Con este método se realiza la medida de la distancia y se guarda en el atributo `_medida`.

Para realizar la medida hay que utilizar la función `analogRead(_salida)` que lee el valor del sensor en el ADC.

Dado que los valores obtenidos no representan la distancia al objeto, se necesita hallar unas expresiones que representen esta distancia en función de la lectura del ADC. Para ello se va a utilizar la grafica en la que se caracteriza al sensor en el apartado 4.2.4.

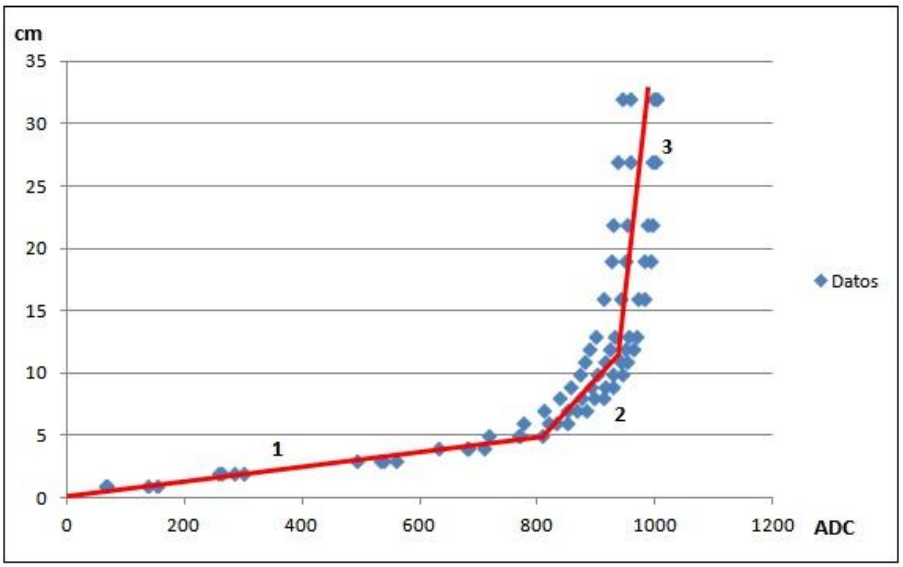


Figura 4.37 Ajuste del comportamiento del sensor infrarrojo a tres rectas

En la grafica anterior se ha representado el comportamiento estático del sensor con tres rectas cuyas formulas son las siguientes:

1. $y = 0.006 \cdot \mathrm{x}$ (Para x menores o iguales que 805).
2. $y = 0.049 \cdot \mathrm{x} - 34.445$ (Para x menores o iguales que 948 y mayores que 805).
3. $y = 0.476 \cdot \mathrm{x} - 439.248$ (Para x mayores que 948).

Finalmente, el valor de `_medida` se calcula utilizando las formulas anteriores en cada caso dependiendo de la entrada del ADC.

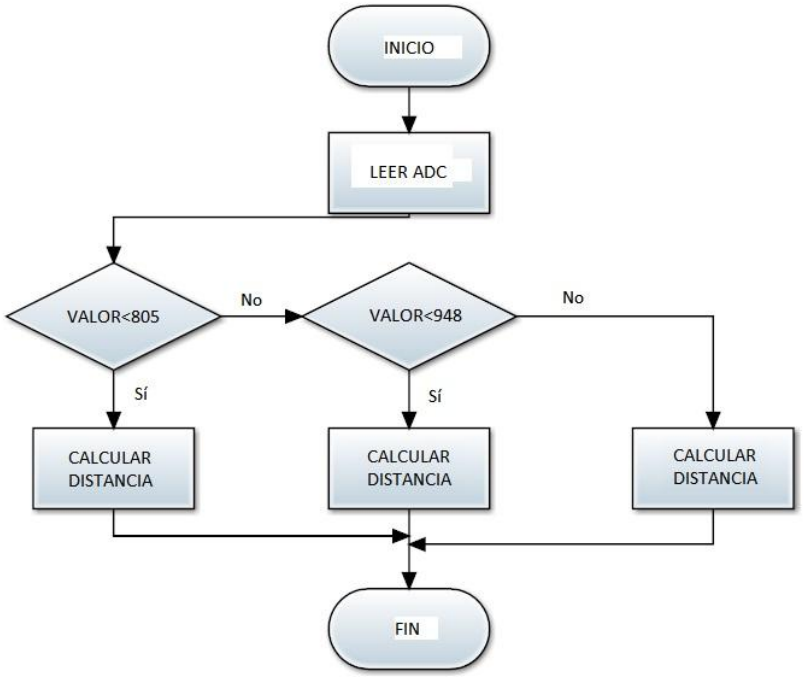


Figura 4.38 Diagrama de flujo del sensor infrarrojo

4.4.4 SensorInclinacion

En el software se ha declarado `SensorInclinacion` como una clase agregada a `Robot`. Los atributos de esta clase son el numero de los pines analógicos a los que se han conectado las salidas del sensor `_x, _y` y `_z,` todas de tipo `byte,` dos variables `float _anguloxy,_anguloxz` donde se guardan las inclinaciones que sufre el acelerómetro.

Los métodos importantes de esta clase son los siguientes.

void conectar (void)

Con este método se configuran los pin analógicos del microcontrolador para poder leer valores a través del convertidor analógico-digital.

En este caso, los pines a los que se conectan las salidas del sensor se requiere que sean de entrada por lo que se declaran como `pinMode(_x/_y/_z, INPUT)`.

void medir (void)

Con este método se realiza la medida de la inclinación y se guarda en los atributos `_anguloxz` y `_anguloxy`.

Para realizar la medida lo primero que se ha de incluir es la función `analogRead(_x/_y/_z)`. Los valores leídos se guardaran en las variables `xVal`, `yVal` y `zVal`, y se cambiaran a una escala de -500 a 500 para poder obtener ángulos de inclinación negativos y así saber si la inclinación es hacia delante o atrás e izquierda o derecha. Para esto se usa la función `map(0,1024,-500,500)`.

Una vez realizado el cambio de escala hay que realizar una serie de cálculos trigonométricos para obtener la formula que represente los ángulos de inclinación en función de los valores leídos.

Estas formulas se implementan utilizando las funciones `atan(xVal/zVal)` y `atan(yVal/zVal)`. Seguidamente se multiplicaran estos valores por `57,2958` para expresar los radianes en grados y obtener los valores de `_anguloxz` y `_anguloyz`.

4.4.5 Motores

En el software se ha declarado la clase `Motores` como una clase agregada a `Robot` y a su vez está compuesta por la clase `Servo`. Esta clase permite controlar los servos del robot, tanto su velocidad como el sentido de giro.

Los principales atributos de esta clase son dos variables tipo `Servo`, `_servoi` y `_servod`, y el número de los pines digitales a los que se han conectado el cable de control de los servos, `_pin1` y `_pin2`.

Los métodos importantes de esta clase son los siguientes.

void conectar (void)

Con este método se configuran los pines digitales del microcontrolador para poder generar una señal PWM que pueda controlar los servos.

En este caso, se utiliza los métodos de la clase `Servo`, `void attach (_pin1)` y `void attach(_pin2)`.

Void mover (int a, int b)

Con este método se controla la velocidad y el sentido de giro de cada servo.

Para ello, se modifica la señal PWM de salida utilizando el método de la clase `Servo`, `void write(float a)` y `void write(float b)` en los que tanto el parámetro a como el b pueden tomar valores de 0 a 180.

Los valores 0 y 180 hacen girar el servo a su máxima velocidad pero en sentidos contrarios mientras que el valor 90 detiene el servo.

4.4.6 Encoder

En el software se ha declarado la clase `Encoder` como una clase agregada a `Robot`.

Esta clase permite controlar los encoders del robot y obtener el numero de pulsos registrados y la velocidad instantánea de giro de las ruedas. Los principales atributos de esta clase son el pin donde está conectado la salida del encoder, `_pin`, y dos variables volátiles, una tipo `int` donde se guardan los pulsos registrados, `_medida`, y otra tipo `float` donde se guarda la velocidad instantánea de giro, `_velocidad`.

Los métodos importantes de esta clase son:

void conectar (void)

Con este método se configura el pin digital del microcontrolador donde se conecta la salida del encoder.

Dado que el conteo de pulsos del encoder se realiza utilizando interrupciones externas de código, se va a incluir la función `attachInterrupt (_pin, medir(), FALLING)`.

void medir (void)

Con este método se podrá contar el número de pulsos que recibe el microcontrolador por el pin establecido y la velocidad de giro de la rueda.

Para ello se utilizara la función `float millis(void)` y la variable `lastmillis`.

Cuando el pin de la interrupción externa lee un flanco de bajada la variable `_medida` aumenta su valor en una unidad. Después se guarda el valor actual de `millis` en la variable `lastmillis`. Cuando el pin vuelve a recibir otro flanco de bajada se realizara una comparación entre el valor de `millis` actual y el de `lastmillis`.

Si la diferencia entre estos dos valores es menor que 100 se habrán recibido dos flancos de bajada en menos de 100 milisegundos lo que es imposible dada la velocidad máxima de giro de la rueda. Por lo tanto se toma ese flanco de bajada como un rebote de la salida del sensor y no se contabiliza como un nuevo pulso. De esta forma se implementa una técnica antirebotes en el encoder.

Si el valor entre pulsos es mayor que 100 milisegundos se contabiliza como un nuevo pulso y se incrementa una unidad el valor de medida.

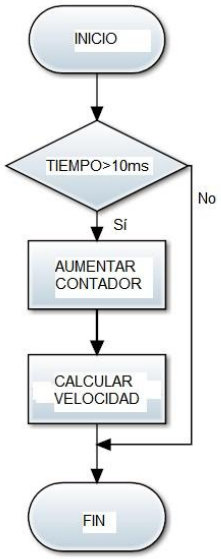


Figura 4.39 Diagrama de flujo del encoder

4.4.7 Comunicacion

En el software se ha declarado la clase `Comunicacion` como una clase agregada a `Robot`. Esta clase permite enviar y recibir los datos necesarios para la comunicación entre el móvil y el microcontrolador. Esta clase utiliza la clase `Serial` que es descrita en el apartado siguiente.

Los métodos más importantes de esta clase son los siguientes.

void enviar (void)

Este método envía toda la información necesaria al modulo Bluetooth para que seguidamente la transmita.

void recibir (void)

Este método evalúa si se recibe algún dato y lo guarda en una variable.

4.4.8 Serial

La clase `Serial` es una clase que ya está incluida en las bibliotecas predefinidas de Arduino pero dada su importancia en la comunicación se ha decidido incluirla aunque no haya sido creada por el autor. Esta clase permite escribir y recibir datos por el puerto serie y se ha declarado como una clase que es utilizada por la clase `Comunicacion`.

Tiene muchos atributos diferentes que no serán descritos en este trabajo, pero se describirán los dos métodos más importantes que se han utilizado en la programación.

void begin (int baudrate)

Inicia la comunicación con el puerto serie con los baudrates deseados.

int read (void)

Devuelve primer byte recibido por el puerto serie. Si no hay datos devuelve -1.

void print (char variable)

Escribe la variable en el puerto serie de salida como un character.

void serialEvent (void)

Ejecuta la función deseada cada vez que haya datos nuevos para leer en el puerto serie.

4.4.9 Navegación

Esta clase recoge todos los métodos necesarios para la navegación del robot como son el control de los motores y el cálculo de la odometría.

Esta clase utiliza la clase `Encoder` y `Motores` ya que entre sus atributos son `_encoderi`, `_encoderd` y `_motores`, haciendo referencia a los dos encoders del robot y a los dos motores. Otros atributos son las coordenadas del robot _x, _y y su orientación `_angulo`.

Los métodos más importantes de esta clase son los siguientes.

void controlmotores (void)

Este método permite regular la velocidad de los motores para que ambas sean iguales.

Para ello se accede a las velocidades de los encoders con el método `float getVelocidad (void)`.

Si la diferencia entre las velocidades de cada rueda es mínima no se realiza ninguna acción ya que se considera un error muy pequeño inapreciable en la trayectoria. Si esta diferencia es mayor se aumenta la velocidad de la rueda más lenta y se reduce la de la más rápida hasta que se estabilicen.

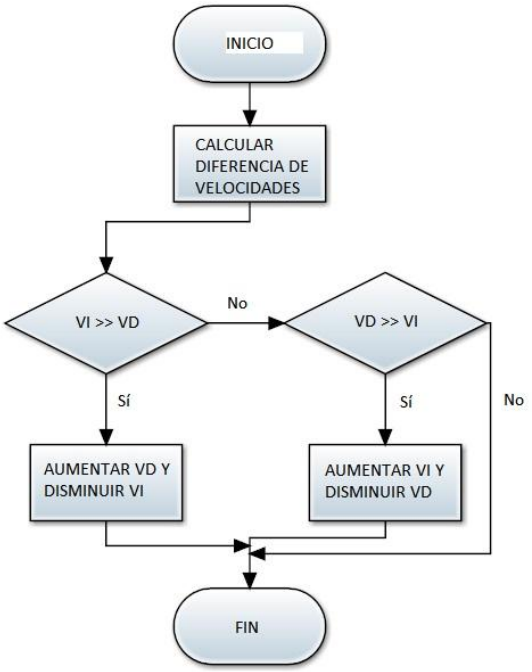


Figura 4.40 Diagrama de flujo del control de la velocidad

void medir (void)

Este método permite calcular la odometría del robot.

Debido a que el encoder implementado no proporciona el sentido de giro del robot, hay que diferenciar el cálculo en 3 estados, avance, giro a la izquierda y giro a la derecha. Esto se realiza a partir de los valores de velocidad que se aplican a los servomotores.

Para cada caso se calcula la distancia recorrida y el ángulo girado utilizando las formulas halladas en el apartado 4.3.

Conocidos estos valores se podrán obtener para todos los casos los valores de _x e _y utilizando las fórmulas 4.4 y 4.5.

Cada vez que se ejecuta este método, al finalizar, se reinician los valores de los pulsos registrados por cada encoder para tomar una medida precisa en la siguiente medida.

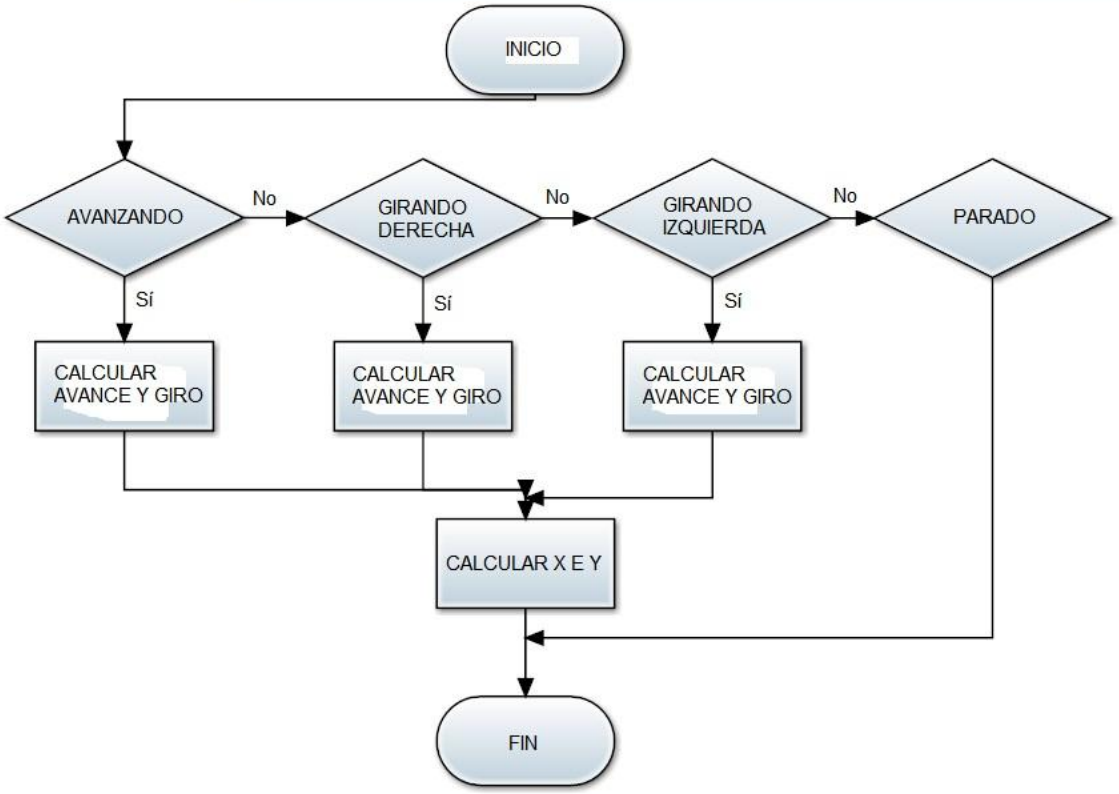


Figura 4.41 Diagrama de flujo del cálculo de la odometría

4.4.10 Robot

Esta clase define las distintas funciones del robot y está formada por el resto de clases anteriormente descritas. Como atributos variables que utilizan las clases descritas anteriormente: `SensorUltrasonidos _sensorultrasonidos`, `SensorInfrarrojos _sensorinfrarrojos`, `SensorInclinacion _sensorinclinacion`, `Encoder _encoderi`, `Encoder _encoderd`, `Motores _motores`, `Navegacion _navegacion` y `Comunicacion _comunicacion`.

Los métodos más importantes de esta clase son los siguientes.

void avanzar (void)

Este método da la orden a los motores de avanzar en línea recta al ejecutar también el método `void controlmotores(void)` de la clase `Navegacion`.

Si el robot encuentra un obstáculo frontal a menos de 15 centímetros, automáticamente se para.

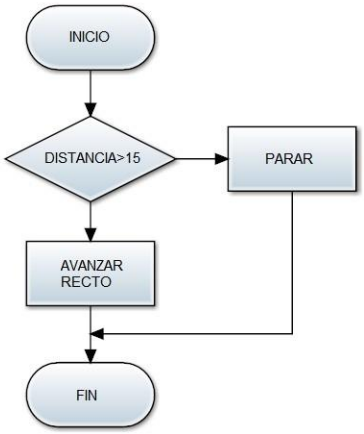


Figura 4.42 Diagrama de flujo del sensor ultrasónico

void girarderecha (int angulo)

Este método recibe el ángulo que se requiere girar y da la orden a los `_motores` de que giren en direcciones opuestas hasta que el sistema de `_navegación` detecte que `_encoderi` y `_encoderd` han girado ese ángulo.

Figura 4.43 Diagrama de flujo de giro a la derecha

void girarizquierda (angulo)

El método es el mismo que el anterior solo que los motores giran en sentido contrario.

void parar (void)

Este método da la orden a los motores de parar.

void esquivar (void)

Este método hace que el robot avance y si encuentra un obstáculo lo esquiva y sigue en la misma dirección que antes de esquivarlo.

Para implementar este algoritmo se ha utilizado una maquina de estados que se muestra en un diagrama a continuación.

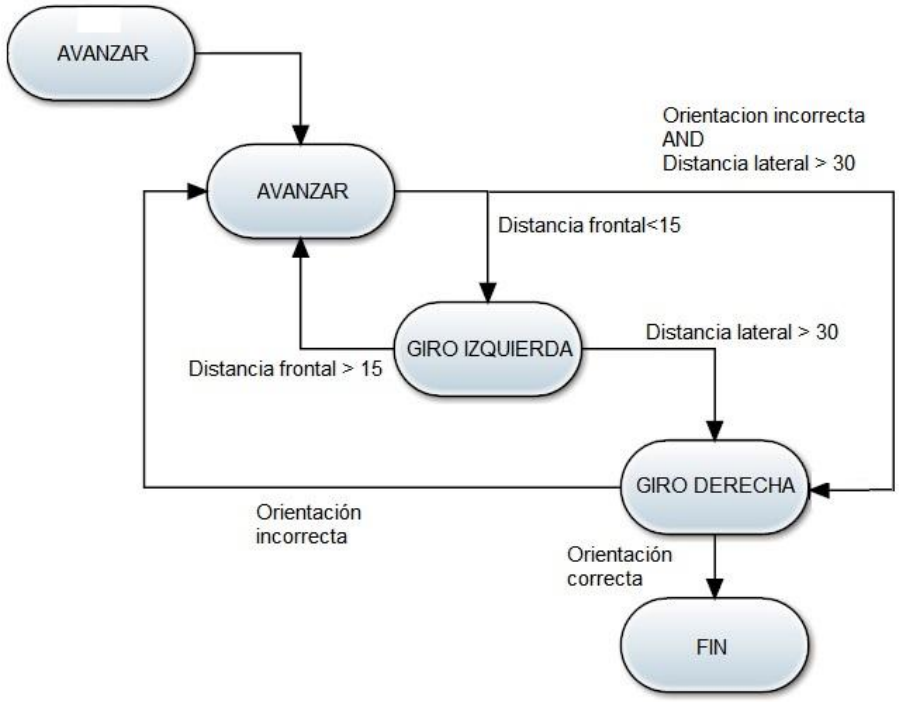


Figura 4.44 Diagrama de estados de evasión de obstáculos

void seguirparedes (void)

Este método hace que el robot sigua la trayectoria de una pared

Para implementar este algoritmo se ha utilizado una maquina de estados parecida a la utilizada en el método `void esquivar (void)` que se muestra el siguiente diagrama.

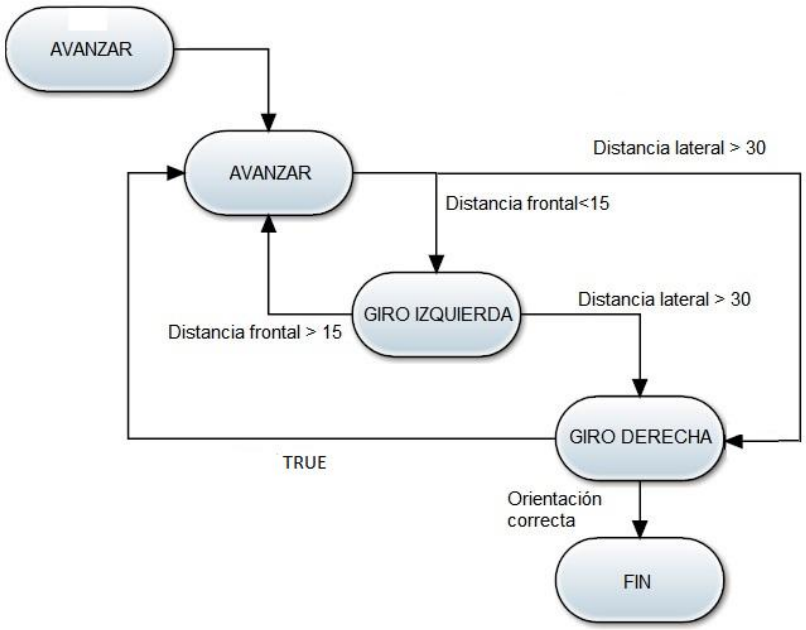


Figura 4.45 Diagrama de estados del seguimiento de paredes

4.4.11 Programa principal

El algoritmo que describe el programa principal que se carga en el robot está basado en una estructura de control de flujo tipo `switch case`. Dependiendo del comando recibido por el puerto serie, se entra a uno u a otro.

Dentro de esta estructura estarán contempladas las siguientes acciones.

1. `robot.avanzar()`
2. `robot.parar()`
3. `robot.girarderecha(45)`

```
4. robot.girarderecha(90)
5. robot.girarizquierda(45)
6. robot.girarizquierda(90)
7. robot.girarizquierda(180)
8. robot.esquivar()
9. robot.seguirparedes()
10.robot.volverinicio()
```

A parte de ejecutar una de estas acciones por cada ciclo de programa, también se ejecutaran las instrucciones para que los sensores tomen sus medidas utilizando los métodos `nombredelsensor.getMedida()` y para enviar la información por Bluetooth utilizando el método `comunicación.enviar()`.

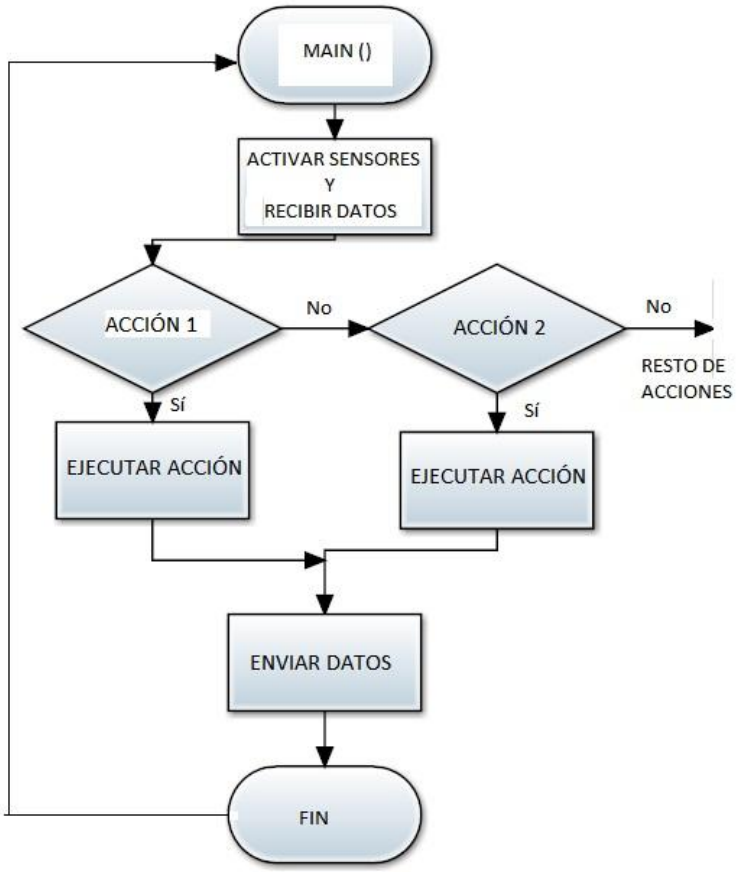


Figura 4.46 Diagrama de flujo del sensor ultrasónico

4.5 Programación de la aplicación móvil

La aplicación móvil se ha desarrollado utilizando la herramienta web App Inventor, esta herramienta ha permitido diseñar una aplicación que cumple las siguientes funcionalidades.

1. Conectarse mediante Bluetooth a un cliente esclavo.
2. Enviar y recibir datos mediante la conexión Bluetooth.
3. Mostrar la información que se recibe por Bluetooth por pantalla.
4. Transformar comandos de voz a texto.
5. Enviar diferentes órdenes al robot según el comando recibido.
6. Mostrar una imagen dinámica por pantalla que representa al robot en un mapa.
7. Trazar en el mapa la trayectoria que sigue el robot.
8. Mostrar en el mapa los obstáculos que detecta el robot.
9. Resetear la posición del robot para ponerla a cero.
10. Guardar la información recibida en un archivo interno del teléfono.

4.5.1 Implementación

La programación de la aplicación se realiza mediante la conexión de diferentes bloques que permiten utilizar las distintas funcionalidades mencionadas, no es necesario escribir un código estructurado utilizando un lenguaje típico de programación de aplicaciones móviles como Java.

Los bloques más importantes utilizados han sido los siguientes:

1. Botones. Permiten realizar una acción al ser pulsados.
2. Labels. Muestran un valor por pantalla en una posición predefinida.
3. Imagesprite. Imagen que se puede cargar en la aplicación.
4. Canvas. Imagen sobre la que se pueden representar figuras
5. Clock. Permite ejecutar una acción cada tiempo predefinido.
6. Speech Recognition. Interpreta el audio que recoge y lo convierte a texto utilizando un algoritmo de STT (Speech To Text).
7. Bluetooth Client. Crea un cliente de Bluetooth maestro que puede conectarse a un esclavo.
8. File1. Permite leer y escribir datos en un fichero interno del teléfono.

4.5.2 La interfaz de usuario

La interfaz de usuario muestra el valor de los sensores en diferentes, el mapa y los botones de comando y reset. En la siguiente imagen se puede ver una captura de la pantalla y los distintos elementos que forman la interfaz.

Figura 4.47 Interfaz de usuario

1. Boton de comando. Al pulsarlo se activara el Speech Recognition que interpretara el comando de voz.
2. Etiqueta de estado de la conexión. Muestra el estado de la conexión(Conectado/desconectado)
3. Distancia frontal. Muestra el valor que recoge el sensor de ultrasonidos.
4. Distancia lateral. Muestra el valor que recoge el sensor de infrarrojos
5. Inclinación frontal.
6. Inclinación lateral.

7. Robot. Imagen esquemática del robot implementada como un Imagesprite de 14x14 pixels.
8. Mapa. Muestra la trayectoria del robot de la en un color dependiente inclinación y los obstáculos que encuentra en su camino. Se implementa con un "Canvas" en blanco.
9. X. Coordenada respecto al punto de inicio.
10. Y. Coordenada respecto al punto de inicio.
11. Angulo Orientación del robot respecto al sistema de referencia fijo del punto de inicio.
12. Distancia recorrida. Distancia total recorrida desde el inicio de la marcha.
13. Velocidad. Velocidad instantánea
14. Botón de reset. Resetea el punto de inicio a las coordenadas actuales.
15. Escala.
16.

4.5.3 La conexión

El primer paso para conectarse será emparejar el teléfono móvil con el modulo Bluetooth del robot.

Una vez emparejado, se pulsa el botón de Comando. Este botón funciona como una interrupción externa y cuando se pulsa, se activa el modo Speech Recognition el cual interpreta el audio recibido y lo convierte a un texto que se almacena en una variable.

Finalmente se evalúa el texto interpretado. Si el comando recibido es CONECTAR entonces se ejecuta la función `bluetoothclient.connect (MAC address)`. El parámetro que recibe es la dirección MAC del esclavo que es 98:D3:31:50:0B:4C.

Si la conexión es exitosa el modulo esclavo envía una confirmación al maestro y se muestra por pantalla en la etiqueta del estado de la conexión la palabra CONECTADO.

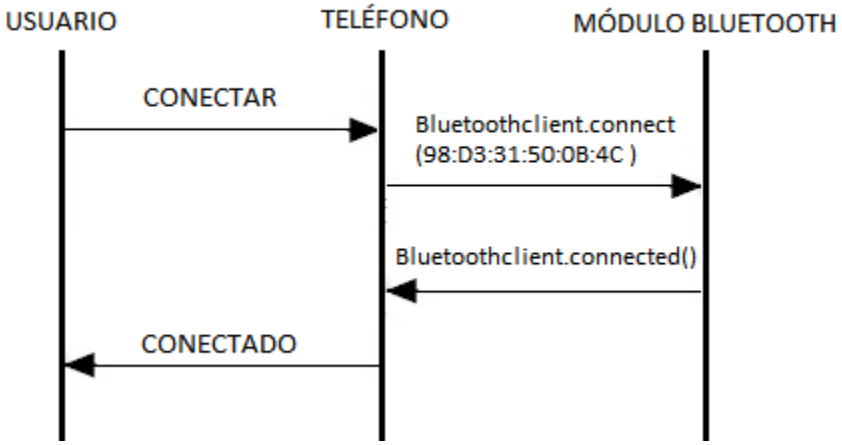


Figura 4.48 Diagrama de secuencia de la conexión Bluetooth

Cada segundo de comprueba si la conexión sigue existiendo y en el caso de que no sea así, se mostrara por pantalla el estado DESCONECTADO.

4.5.4 Comandos de voz

Cada vez que se recibe un comando de voz, el modo Speech Recognition lo interpreta y lo convierte a texto utilizando algoritmos STT.

Seguidamente este texto se evalúa para ver si coincide con alguna de las órdenes predefinidas del programa.

Estas son las siguientes:

1. "Avanza".
2. "Para".
3. "Derecha". Gira 90 grados a la derecha.
4. "Uno". Gira 90 grados a la derecha.
5. "Izquierda". Gira 90 grados a la izquierda.
6. "Dos". Gira 45 grados a la izquierda.
7. "Media vuelta".
8. "Esquiva".
9. "Paredes".

Cada orden tiene asignado un número del 1 al 9 y este número se envía por Bluetooth para que lo interprete el microcontrolador. Para enviar el dato se utiliza la función `bluetoothclient1.send1bytenumber(numero)`.

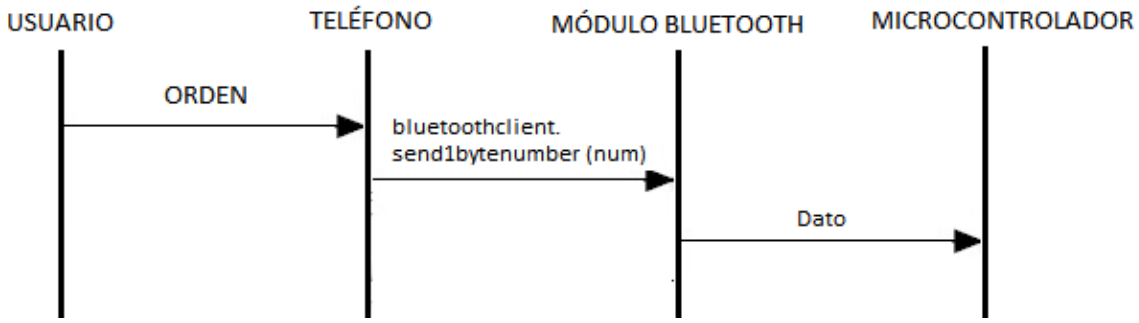


Figura 4.49 Diagrama de secuencia de el envío de instrucciones

Como se puede observar, este tipo de programación facilita mucho añadirle más acciones al robot desde la aplicación, ya que solo habría que añadirle un comando más.

4.5.5 Adquisición de datos

La aplicación debe ser capaz de recibir los datos que le envía el microcontrolador por Bluetooth y almacenarlos para su uso posterior.

Se configura un evento cada 300 milisegundos que compruebe si existen datos de entrada en el puerto serie. Para ello se utiliza la función `bluetoothclient1.bytesavailabletoreceive()` que devuelve -1 si no hay datos disponibles.

Una vez recibidos los datos en el teléfono, se separan por los espacios y los valores se almacenan en un vector de 11 posiciones ya que son la cantidad de datos que se espera recibir. Estos son:

1. `Distancia frontal`. Distancia en centímetros al obstáculo frontal.
2. `Distancia lateral`. Distancia en centímetros al obstáculo lateral.
3. `Inclinación frontal`.
4. `Inclinación lateral`.

5. x. Coordenada según el sistema de referencia fijo.
6. y. Coordenada según el sistema de referencia fijo.
7. `Orientación`. Angulo de orientación según sistema de referencia fijo.
8. `Distancia recorrida`. Distancia recorrida entre medida y medida.
9. `Velocidad`. Velocidad instantánea.
10. `Inclinación`. Grado de inclinación de 1 a 3.
11. Signo de llegada "/".

Seguidamente se evalúa si estos datos han llegado correctamente utilizando el signo de llegada "/" y si no es así se descartan.

Si han llegado correctamente se representan sus valores por pantalla en las distintas Labels que se han configurado para ello.

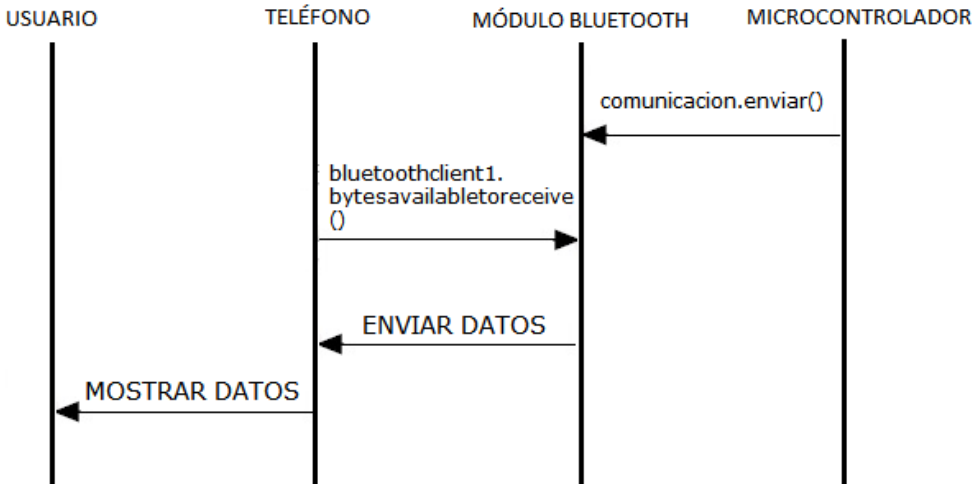


Figura 4.50 Diagrama de secuencia del recibimiento de datos

Todos los datos que han llegado correctamente se almacenan en el archivo "datos.txt" que se encuentra en la memoria del teléfono.

4.5.6 Representación del mapa

Con la información recopilada por el robot se puede representar un mapa esquemático que muestre su posición, su trayectoria, la inclinación del suelo y los obstáculos que existen. Este mapa se representa en el Canvas.

Posición, orientación y trayectoria.

El robot es representado por un Imagesprite que se carga en el programa previamente la cual tendrá unas dimensiones de 14x14 pixels. Su posición inicial del centro será en las coordenadas `(x, y)` del Canvas.

Esta figura puede posicionarse por la pantalla a voluntad utilizando la función `imagesprite1.moveto(x, y)` por lo que , conocidos los valores reales de x e y que se reciben por Bluetooth, se posiciona el centro de la imagen cada vez que estos se actualizan.

La relación entre pixeles y distancia real es de 1 a 1, es decir, un pixel representa un centímetro de la distancia recorrida real. Esta escala puede ser modificada posteriormente y se explica en un apartado posterior.

Para representar la orientación del robot se utiliza la función `image.sprite1.headingto(Giro)` la cual permite rotar la imagen para orientarla hacia el ángulo real del robot que se recibe por Bluetooth.

Para representar la trayectoria que sigue el robot se utiliza la función `canvas1.drawline(x1, y1, x2, y2)` donde las coordenadas con índice 1 representan el punto inicial y las de índice 2 representan el punto final.

El color de esta línea puede definirse en función de la variable `Inclinacion`. La línea será verde cuando esta sea 0, amarilla cuando esta es 1 y roja cuando sea 2. De este modo se puede realizar una estimación visual del grado de inclinación del terreno a lo largo de la trayectoria.

Representación de obstáculos

Los diferentes obstáculos que pueden aparecer en el mapa se representan como puntos y para ello hay que hallar sus coordenadas.

Como se conoce la distancia a los obstáculos y la posición del robot, se podrán calcular por trigonometría.

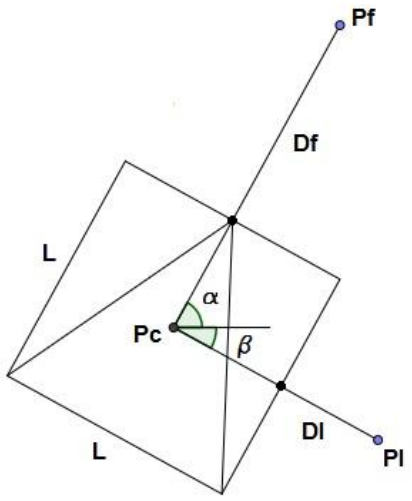


Figura 4.51 Diagrama de representación de obstáculos

Según la figura las coordenadas del los puntos Pf (Punto frontal) y Pl (Punto lateral) quedan definidas en función del punto central Pc, el ángulo α, el lado del cuadrado L y las distancias frontales y laterales Df y Dl respectivamente.

$$Pfx = Pcx + \left(\frac{L}{2} + Df\right) x \cos(\alpha) \quad [4.12]$$

$$Pfy = Pcy + \left(\frac{L}{2} + Df\right) x \operatorname{sen}(\alpha) \quad [4.13]$$

Conociendo que $\beta = 90 - \alpha$. Entonces:

$$Plx = Plx + \left(\frac{L}{2} + Dl\right) x \cos(90 - \alpha) \quad [4.14]$$

$$Ply = Ply + \left(\frac{L}{2} + Dl\right) x \operatorname{sen}(90 - \alpha) \quad [4.15]$$

4.5.7 Reset

A medida que nuevos datos llegan por Bluetooth, la posición, orientación y obstáculos se van actualizando. Pero existe el botón Reset que permite al usuario devolver al robot a su posición inicial en el mapa y borrar toda la información que haya en el.

Al pulsar este botón se envía una orden al robot para que reinicie toda su odometría y establezca su posición actual como posición inicial.

4.5.8 Escala

El usuario puede introducir mediante el teclado el valor de la escala deseada. Este valor equivale a la distancia real en centímetros que correspondiente a un pixel de la imagen.

A partir de ese momento todas las representaciones gráficas en el mapa tendrán esa escala hasta que se vuelva a cambiar.

Capitulo 5: Experimentación

En este capítulo se exponen las diferentes pruebas llevadas a cabo en el robot y en la aplicación móvil para evaluar su funcionamiento así como un análisis de los resultados obtenidos.

Para ello, se evalúa la información recopilada por los sensores y los diferentes movimientos puede realizar el robot. La trayectoria definida por el robot y su posición se van a comparar con la representación en el mapa de la aplicación y se comprobara si se corresponden.

5.1 Pruebas de sensores

Para evaluar el funcionamiento de los sensores se realizaran distintas pruebas en las que se comprobara el valor real con el valor medido.

5.1.1 Sensor de ultrasonidos

El sensor de ultrasonidos mide la distancia frontal desde el sensor al obstáculo. Para evaluar su funcionamiento se realizaran 10 medidas desde 5 a 120 centímetros con distintos ángulos; 0, ± 15° y ± 30°.

En la siguiente gráfica se muestran los resultados obtenidos, en el eje de ordenadas la distancia real, en el eje de abscisas, la distancia medida.

Figura 5.1 Comportamiento del sensor ultrasónico

En la gráfica se puede observar que las medidas obtenidas para un rango total de 60º son muy precisas hasta los 50 centímetros aproximadamente. A partir de esta distancia el rango se reduce a 30º hasta los 90 centímetros ya que a partir de ahí se comienzan a apreciar ligeras desviaciones si el obstáculo no está a 0º.

En general el sensor tiene un comportamiento muy bueno si el obstáculo se encuentra en su línea frontal de visión.

5.1.2 Sensor de infrarrojos

Este experimento es de gran importancia ya que se pretende apreciar si la aproximación que se ha hecho utilizando la fórmula del apartado 4.4.2 es fiable o no.

Para evaluar su funcionamiento se han realizado 10 medidas desde 1 a 32 centímetros con luminosidades externas diferentes: luz natural, luz artificial y sin luz.

En la siguiente gráfica se muestran los resultados obtenidos, en el eje de ordenadas la distancia medida, en el eje de abscisas, la distancia real.

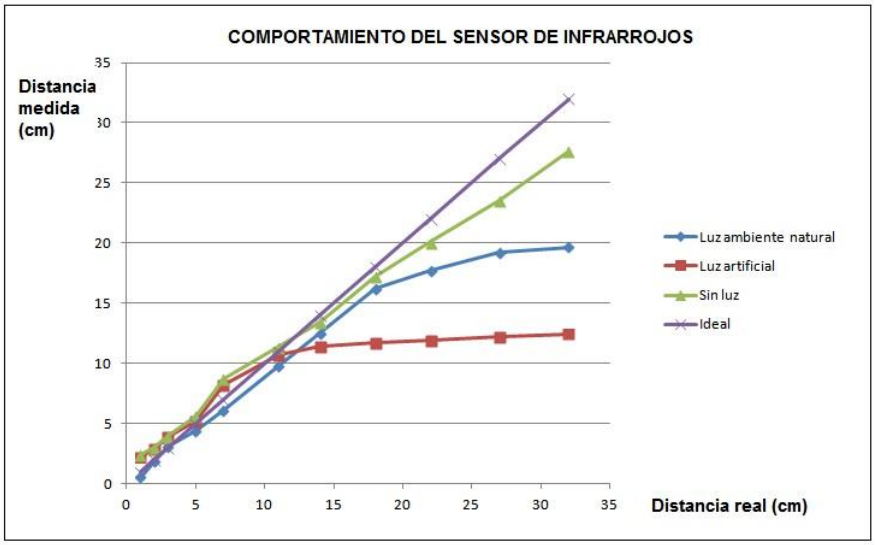


Figura 5.2 Comportamiento del sensor de infrarrojos

Se puede observar que los valores medidos para todas las luminosidades son muy precisos para distancias menores que 12 centímetros. A partir de esa medida los datos se vuelven inexactos.

5.1.3 Acelerómetro

El acelerómetro mide la inclinación a la que está sometido el robot, para evaluar el comportamiento del sensor se colocara al robot en una plataforma con ángulos de inclinación conocidos de 0 a 20 grados hacia la derecha, hacia la izquierda, hacia delante y hacia detrás.

Los resultados se exponen en las siguiente gráfica.

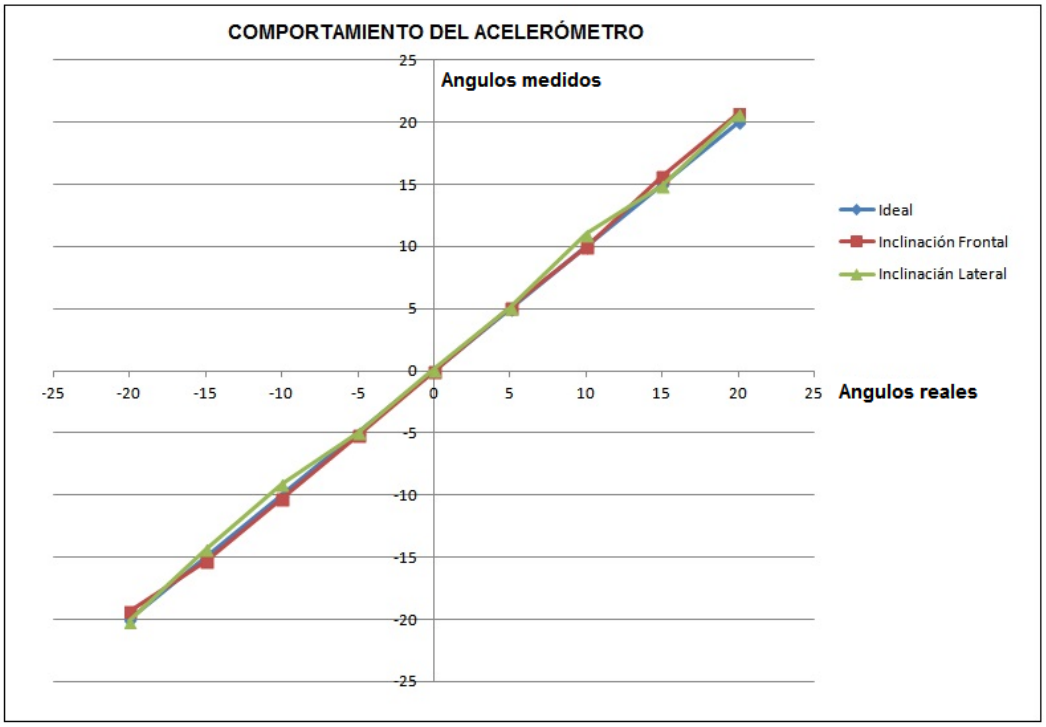


Figura 5.3 Comportamiento del acelerómetro

En esta gráfica se puede observar que el comportamiento del acelerómetro es muy bueno porque los valores medidos son casi idénticos a los reales.

5.2 Pruebas de navegación

Para evaluar los distintos movimientos que realiza el robot se evaluara su comportamiento esperado y la representación grafica que se obtiene de su trayectoria y los obstáculos detectados en el mapa de la aplicación

5.2.1 Movimiento rectilíneo y parada automática

En este apartado se evalúa la capacidad del robot de moverse en línea recta y de pararse antes de chocar contra un obstáculo. Los resultados de este experimento ofrecen una visión precisa del correcto funcionamiento del algoritmo de detección de obstáculos y del sistema de control de los motores implementado.

Figura 5.4 Obstáculo del experimento

La distancia a la que se ha programado la parada automática del robot es a 15 centímetros del obstáculo. Para llevar a cabo el experimento se mide la distancia real a la que el robot se detiene.

Figura 5.5 Representación de la trayectoria 1

En la aplicación móvil se puede observar que el robot avanza en línea recta y tras detectar el obstáculo se para. Se puede comprobar que la distancia al objeto es 2,74 centímetros menor que la requerida. Aunque es un buen valor, esto puede ser debido a que el robot no frena en seco.

5.2.2 Esquivar obstáculos

En este apartado se evalúa si el robot esquiva el obstáculo de manera correcta. El comportamiento esperado es que el robot esquive el obstáculo y continúe la marcha en el sentido que tenía antes de esquivarlo.

Los resultados son los siguientes.

Figura 5.6 Representación de la trayectoria 2

Se puede observar que el robot evita el obstáculo. Los sensores de ultrasonidos e infrarrojo toman las medidas y estas se van representando con bastante fidelidad a medida que el robot avanza. Se ha decidido representar las medidas capturadas por el sensor de ultrasonidos si son menores a 30

centímetros y las del sensor de infrarrojos si son menores que 10 centímetros para eliminar posibles errores.

5.2.3 Seguir paredes

En este apartado se evalúa como sigue las paredes el robot.

El comportamiento esperado es que el robot siga la cara de una pared evitando chocarse y que la información representada en el mapa de la aplicación móvil se ajuste a la realidad.

Para llevar a cabo este experimento este se realizara en una habitación con el mismo obstáculo que el apartado anterior, si el comportamiento es el adecuado el robot debería bordear el obstáculo.

Los resultados son los siguientes.

Figura 5.7 Representación de la trayectoria 3

Se puede observar que el robot sigue la superficie de las paredes y representa los puntos con bastante fidelidad y Las coordenadas recibidas en el punto final

son muy similares a las esperadas aunque se puede observar un ligero error en la coordenada X que puede ser causado por que se hayan contado pulsos de mas en el encoder.

Capítulo 6: Gestión del proyecto

Dado que este proyecto conlleva el diseño y la fabricación de un robot móvil de manera práctica. Se ha decidido incluir un capitulo en el que se describan las tareas que han sido necesarias para llevarlo a cabo y sus tiempos.

Uno de los objetivos de este trabajo es que el robot diseñado tuviera el menor coste posible. Es por eso que el lector encontrara en el último apartado el desglose de todos los materiales utilizados y su precio.

6.1 Descripción de las fases del proyecto

En este apartado se describen las tareas que han sido necesarias en este trabajo y se acompañan de un diagrama de Gant donde se ordenan cronológicamente.

Planteamiento del problema

Esta es la tarea inicial en la que se planteo el campo donde desarrollar el trabajo y el problema a resolver.

Recopilación de la información e investigación

Seguidamente se empezó a buscar publicaciones, trabajos, productos comerciales relacionados con el trabajo que se iba a realizar.

Establecimiento de objetivos

Una vez conocido el estado del arte, se plantearon unos objetivos en base a todo el marco de información recopilado.

Estudio del sistema y definición de requisitos

Antes de empezar a construir el sistema hubo que estudiar sus principales características y definir sus funcionalidades y requisitos. En base a eso se diseño su arquitectura general.

Diseño y fabricación mecánica

En esta fase se realizo un modelo 3D del robot antes de fabricarlo y una vez estuvo realizado se fabrico en base a dicho modelo.

Implementación del hardware

Una vez se disponía de la estructura del robot se tuvieron que seleccionar y comprar los diferentes componentes electrónicos que se iban a incluir. Posteriormente se conectaron y se montaron en la estructura del robot.

Programación del microcontrolador

Tras haber finalizado la estructura del robot se programó el microcontrolador y se crearon las librerías necesarias.

Programación de la aplicación

Antes de programar la aplicación fue necesario realizar un estudio en profundidad de las diferentes herramientas que existían. Una vez se tuvo que aprender a utilizar para llegar a desarrollar a la aplicación definitiva.

Experimentación

En esta fase se desarrollaron diferentes experimentos que evaluarían el comportamiento del robot y su correcto funcionamiento.

Redacción de la memoria

Esta tarea se realizo paralelo junto a todo el resto para ir documentando lo que se iba desarrollando.

6.1.1 Diagrama de Gant

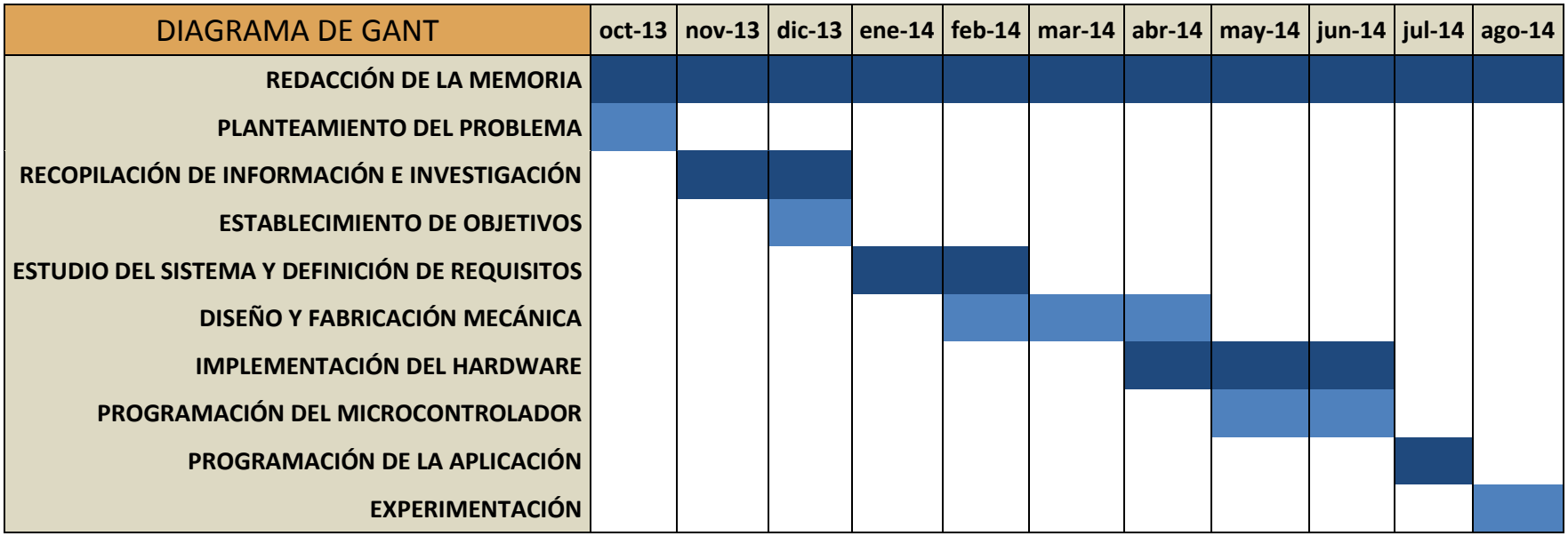


Figura 6.1 Diagrama de Gant

6.2 Presupuesto

En la siguiente figura se muestra el coste del robot incluyendo todas sus piezas. No se incluye el coste de las herramientas ni el de mano de obra.

CONCEPTO	CANTIDAD	PRECIO UNIDAD	TOTAL
Contrachapado de pino (60x30x4)	1	3,90 €	3,90 €
Varilla de madera de pino (60x1,5)	1	1,10 €	1,10 €
Listón de madera de pino (60x2x1)	1	1,10 €	1,10 €
Tornillo M3	8	0,05 €	0,40 €
Tornillo M2 con tuerca	4	0,10 €	0,40 €
Pegamento instantáneo	1	4,70 €	4,70 €
Cola adhesiva	1	3,47 €	3,47 €
Spray pintura azul	1	2,60 €	2,60 €
Ruedas	2	1,90 €	3,80 €
Rueda omnidireccional	1	2,50 €	2,50 €
Servomotor	2	6,30 €	12,60 €
Fotointerruptor	2	1,50 €	3,00 €
Arduino UNO	1	13,90 €	13,90 €
Sensor Ultrasonidos	1	5,10 €	5,10 €
Acelerómetro	1	4,30 €	4,30 €
Modulo Bluetooth	1	3,90 €	3,90 €
Protoboard	2	0,49 €	0,98 €
Cables Macho-Hembra (10uds)	1	1,90 €	1,90 €
Cables Hembra-Hembra (10uds)	1	1,90 €	1,90 €
LED infrarrojo	1	0,49 €	0,49 €
Fotodiodo infrarrojo	1	0,49 €	0,49 €
Resistencias 10 Kohm (5uds)	1	0,90 €	0,90 €
Resistencias 68 ohm (5uds)	1	0,90 €	0,90 €
Condensador 0.33uF	2	0,50 €	1,00 €
LM7806	1	0,90 €	0,90 €
Batería 7,4V 1000mAh	1	6,35 €	6,35 €
			82,58 €

Figura 6.2 Presupuesto

Capítulo 7: Conclusiones y trabajos futuros

Conclusiones generales

En este trabajo se han presentado un robot móvil con capacidad de guiado autónomo y controlado por voz. El trabajo es el resultado de meses de aprendizaje e investigación en los dominios de diseño mecánico de robots, navegación autónoma y reconocimiento de comandos de voz.

Se enmarca en proyectos que tienen entre otros objetivos la aplicación de las técnicas mencionadas al diseño de robots con objetivos educacionales y académicos, que introduzcan a los estudiantes al amplio mundo de la robótica con una mención al diseño de aplicaciones para teléfonos móviles. Es por ese motivo por el que se ha decidido utilizar en este proyecto tecnologías Open-Source con grandes comunidades de usuarios que las respaldan y las expanden día a día.

En relación a la parte más técnica de este trabajo, se han conseguido los objetivos que se habían fijado en un principio.

El resultado de esto es un robot móvil que el usuario puede controlar con una serie de comandos de voz y puede visualizar la información que este capta de su entorno a través de la pantalla de su teléfono móvil. Estos comandos de voz le permiten a robot realizar acciones como avanzar o girar pero también otras que el robot ejecuta de manera autónoma como esquivar obstáculos y seguir un camino.

Las aplicaciones en el ámbito educacional de este robot son muy extensas ya que todos los códigos se han descrito en este trabajo y se harán públicos en internet, permitiendo a todo el mundo conocer el funcionamiento del robot y modificar lo que requiera para mejorarlo o adaptarlo a sus necesidades.

Trabajos futuros

A continuación se sugieren una serie de ideas para mejorar este trabajo que han ido surgiendo durante su desarrollo.

- Mejorar el sistema de percepción con más sensores para ampliar su rango.
- Utilizar la información captada por los sensores para crear un mapa interno con el que el robot pueda orientarse mejor utilizando métodos de SLAM y realizar una navegación más eficiente utilizando algoritmos de planificación de trayectorias.
- Utilizar el mapa de la aplicación para que al pulsar un punto de este el robot se dirija a él.
- Implementar una cámara en el robot y poder que el usuario pueda ver el entorno a través del móvil.
- Mejorar el rango de distancia de las comunicaciones para poder operarlo en entornos exteriores extensos.
- Implementar un comando de retorno al punto inicial.

Bibliografía

[1] Roland Siegwart, Illah R. Nourbakhsh, Introduction to Autonomous Mobile Robots, MIT Press, 2004.

[2] Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F., & Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. Robotics and Automation, IEEE Transactions on, 17(3), 229-241.

[3] Latombe, J. C. (1996). Robot Motion Planning, Chapter.

[4] Robotchallenge-http://www.robotchallenge.org

[5] Renesas-http://www.renesas.eu

[6] Sklar, E., Eguchi, A., & Johnson, J. (2003, January). RoboCupJunior: learning with educational robotics. In RoboCup 2002: Robot Soccer World Cup VI (pp. 238-253). Springer Berlin Heidelberg.

[7] Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. Computers & Education, 58(3), 978-988.

[8] Arduino-http://www.arduino.cc

[9] D'Ausilio, Alessandro. June 01, 2012. Arduino: a low-cost multipurpose lab equipment. Behavior research methods 44, no. 2.

[10] Javier Jimenez, Francisco. September 01, 2014. API for communication between Labview and Arduino UNO. Revista IEEE América Latina 12, 6.

[11] Maritz, Jacques. January 01, 2013. Arduino and MATLAB make scientific contact/Arduino en MATLAB maak wetenskaplike kontak. Suid-Afrikaanse tydskrif vir natuurwetenskap en tegnologie 32, no. 1

[12] Srikanth, Ch. February 25, 2014. Smart Embedded Medical Diagnosis using Beaglebone Black and Arduino. International journal of engineering trends and technology 8, no. 1.

[13] Araújo, A., Portugal, D., Couceiro, M. S., & Rocha, R. P. (2013, April). Integrating Arduino-based educational mobile robots in ROS. In Autonomous Robot Systems (Robotica), 2013 13th International Conference on (pp. 1-6). IEEE.

[14] Balogh, R. Educational Robotic Platform based on Arduino.

[15] Barber, R., De La Horra KÖllmer, M., & Crespo, J. (2013, August). Control Practices Using Simulink with Arduino As Low Cost Hardware. In Advances in Control Education (Vol. 10, No. 1, pp. 250-255).

[16] Roy, M.Prathyusha , K. S. April 25, 2013. Voice and Touch Screen Based Direction and Speed Control of Wheel Chair for Physically Challenged Using Arduino. International journal of engineering trends and technology 4, no. 4.

[17] Mahendran.N. February 25, 2013. Multiple Sensor Feeding Supported Building Automation System Using Arduino Platform With Exposure of 802.15.4 Functionalities. International journal of engineering trends and technology 4, no. 2.

[18] Paralax-http://www.parallax.com

[19] Ruiz, R. L., Vásquez, J. A. Q., & Londoño, G. A. H. (2004). Implementación del protocolo Bluetooth para la conexión inalámbrica de dispositivos electrónicos programables. Scientia et Technica, 1(24).

[20] Ierache, J., Bruno, M., Dittler, M., & Mazza, N. (2008). Robots y Juguetes Autónomos una Oportunidad en el Contexto de las Nuevas Tecnologías en Educación. In JIISIC (pp. 371-380).

[21] Möckel, R., Jaquier, C., Drapel, K., Dittrich, E., Upegui, A., & Ijspeert, A. (2006). YaMoR and Bluemove—an autonomous modular robot with bluetooth interface for exploring adaptive locomotion. In Climbing and Walking Robots (pp. 685-692). Springer Berlin Heidelberg.

[22] Aroca, R. V., Péricles, A., de Oliveira, B. S., Marcos, L., & Gonçalves, G. (2012). Towards smarter robots with smartphones. In 5th Workshop in Applied Robotics and Automation, Robocontrol.

[23] Aroca, R. V., Péricles, A., de Oliveira, B. S., Marcos, L., & Gonçalves, G. (2012). Towards smarter robots with smartphones. In 5th Workshop in Applied Robotics and Automation, Robocontrol.

[24] Nasereddin, H. H., & Abdelkarim, A. A. (2010). Smartphone Control Robots Through Bluetooth. International Journal of Research and Reviews in Applied Sciences, 4(4).

[25] Aram, Siamak, Amedeo Troiano, and E. Pasero. "Environment sensing using smartphone." Sensors Applications Symposium (SAS), 2012 IEEE. IEEE, 2012.

[26] 26Roca Soler, J. (2013). Teleoperación y control de un robot Lego a través de un terminal Android.

[27] 27Bake Fajardo, M. G., & Moreno Godoy, H. G. (2013). Diseño de un robot móvil controlado por una tablet y programación en android que permita la comunicación por bluetooh (Doctoral dissertation).

[28] App Inventor-http://www.appinventor.mit.edu

[29] Wolber, D. (2011, March). App inventor and real-world motivation. InProceedings of the 42nd ACM technical symposium on Computer science education (pp. 601-606). ACM.

[30] J. C., LÓPEZ, J. A., & OROZCO, A. F. Diseño y desarrollo de un Goniómetro basado en Sensores Cinemáticos con comunicación inalámbrica en tiempo real.

[31] SJDavis Robotics-http://www.sjdavisrobotics.com

[32] Arcbotics-http://www.arcbotics.com

[33] Furui, S., Kikuchi, T., Shinnaka, Y., & Hori, C. (2004). Speech-to-text and speech-to-speech summarization of spontaneous speech. Speech and Audio Processing, IEEE Transactions on, 12(4), 401-408.

[34] Delgado Gallardo, V. A., Trujillo Castellanos, A., & Valdez Simón, D. (2014). Sistema de domótica controlado por voz para personas con deficiencias motrices.

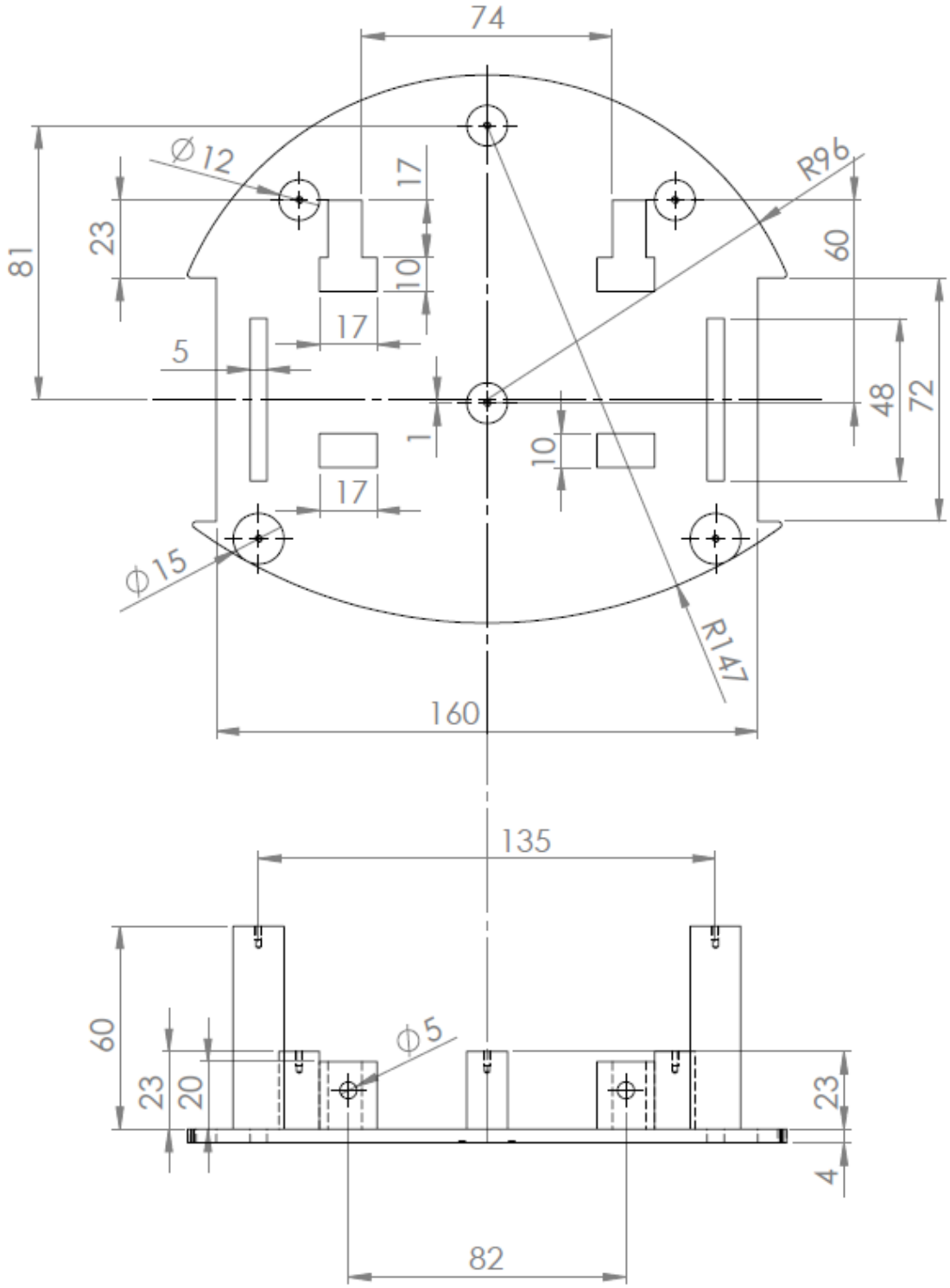
[35] Alsmadi, O. M., Al Jallad, A. A., Abo-Hammoud, Z. S., & Al Majali, F. J. Arduino-Based Automatic Safety Vehicle Control.

[36] Alcubierre, J. M., Minguez, J., Montesano, L., Montano, L., Saz, O., & Lleida, E. (2005, November). Silla de ruedas inteligente controlada por voz. In Primer Congreso Internacional de Domótica, Robótica y Teleasistencia para Todos. Madrid (España).

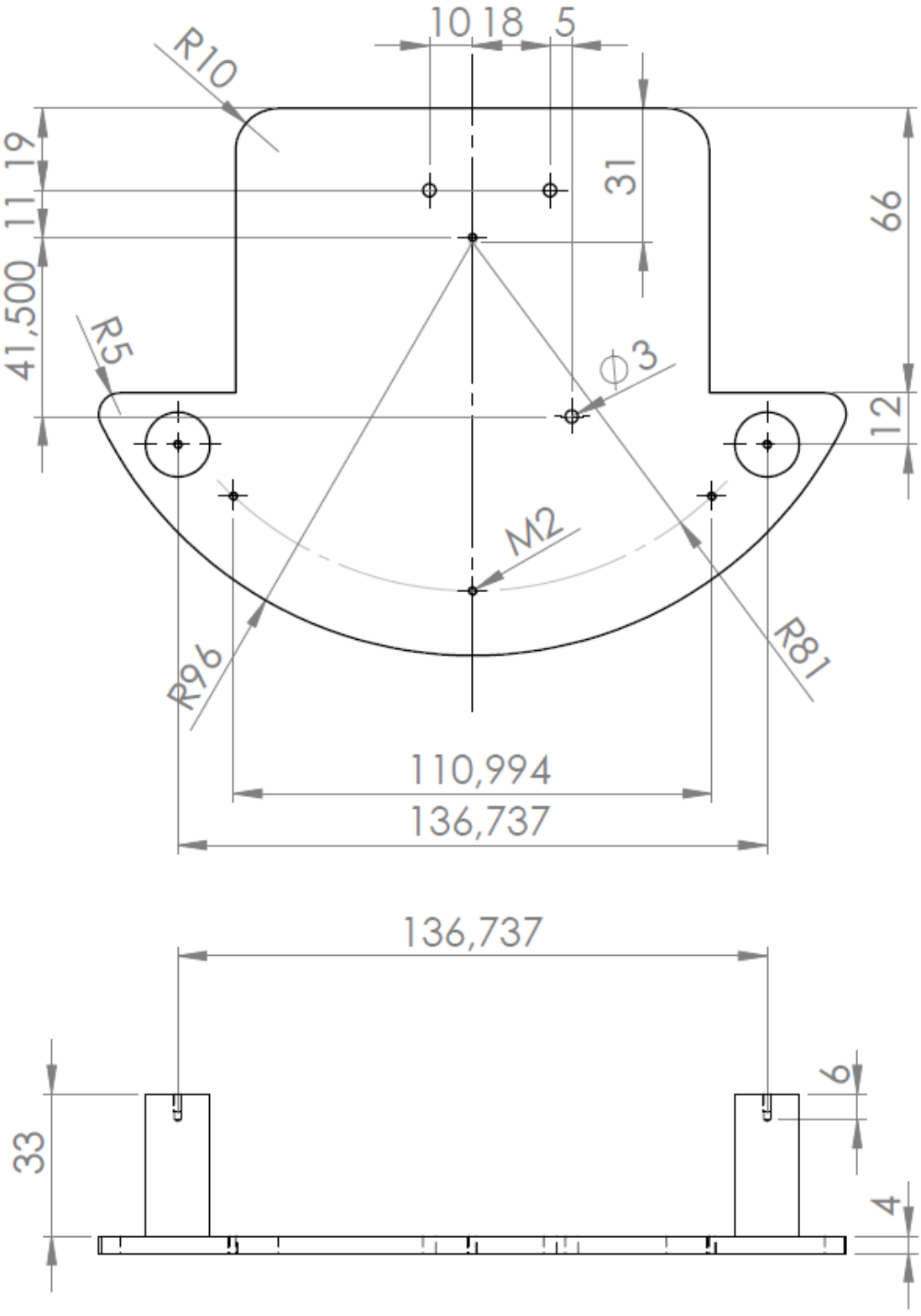
[37] Veear-http://www.veear.eu

Anexos

Planos de la base



Planos de la plataforma intermedia



Planos de la cubierta